

**UNIVERSIDAD AUTONOMA DE MADRID**

**ESCUELA POLITECNICA SUPERIOR**



# **PROYECTO FIN DE CARRERA**

## **Ingeniería de Telecomunicación**

**Aplicación iOS de gestión mecánica de vehículos**

**Pablo Zarco Sanz**

**Noviembre 2015**



# **Aplicación iOS de gestión mecánica de vehículos**

**AUTOR: Pablo Zarco Sanz**  
**TUTOR: Juan José Sánchez Peña**  
**PONENTE: Manuel Sánchez-Montañés Isla**

**Escuela Politécnica Superior**  
**Universidad Autónoma de Madrid**  
**Noviembre de 2015**



## ***Resumen***

Este proyecto consiste en el desarrollo de una aplicación para dispositivos móviles basada en el entorno operativo iOS. La aplicación está orientada al análisis de gastos y gestión de mecánica de vehículos, aunque es exportable a otros medios de transporte. Dispone de diversas herramientas visuales que pueden ayudar a conductores a hacer un seguimiento fácil de sus vehículos.

Adicionalmente consta de un sistema de gestión de documentos donde el usuario podrá almacenar una copia de toda aquella documentación obligatoria del vehículo.

la aplicación esta desarrollada para el parque automovilístico español y se encuentra publicada en App Store de Apple.

## ***Palabras clave***

Vehículos, gestión, análisis, aplicación, iOS, Dropbox, Documentos, Localización.

***Abstract***

This project consists of an iOS mobile phone application development. This project is oriented to drivers with low understanding of his vehicle. It has a huge range of options which can guide users easily into the world of mechanics.

The application also has storage services where users can save a copy of their vehicle documents and keep them safely, including the option of Dropbox synchronization.

The application is developed for the Spanish market and it is now available in the Apple App Store.

***Key Words***

Vehicles, storage, review, application, iOS, Dropbox, Dropbox, Localization.

## ***Agradecimientos***

En primer lugar y de manera muy especial, quiero agradecer a mis padres, mi hermano y toda mi familia por el apoyo que me han dado durante estos años. Sin ellos habría sido imposible llegar hasta aquí.

A Irene, que me ha dado fuerzas tanto en los buenos como en los malos momentos para seguir.

Agradecer a Juan José y a Manuel por darme la oportunidad de realizar este proyecto.

Y por último a todos los amigos y compañeros de la escuela que me han ayudado en este feliz periodo de mi vida.

## ÍNDICE

1 INTRODUCCIÓN	1
1.1 Motivación.	1
1.2 Objetivos.	2
1.3 Metodología de trabajo.	3
1.4 Organización de la memoria.	3
2 ESTADO DEL ARTE	4
2.1 Introducción.	4
2.2 Apple Inc.	5
2.2.1 iPhone y App Store.	5
2.2.3 Objective-C.	6
2.3 Análisis de aplicaciones similares.	6
2.3.1 Vehic.	6
2.3.2 Mi Coche.	8
2.3.3 Car Controller.	9
2.3.4 uCar.	10
2.3.5 Causas comunes en la eliminación de aplicaciones.	11
2.3.6 Conclusiones.	12
3 DISEÑO	13
3.1 Objeto del proyecto.	13
3.2 Diagrama de estados.	14
3.3 Análisis de la aplicación.	15
3.3.1 Garaje.	15
3.3.2 Historial.	16
3.3.3 Usuario.	16
3.3.4 Documentos.	16
3.3.5 Matrículas.	17
3.3.6 Distancias.	17
3.4 Entorno de la aplicación.	17
3.4.1 Sistema operativo.	17
3.4.2 Requisitos del sistema operativo.	20
3.4.3 El problema de las pantallas.	21
3.4.4 Requisitos de la aplicación.	22
3.5 Limitaciones	22
3.5.1 Limitaciones de lenguaje.	22
3.5.2 Limitaciones técnicas	22
3.5.3 Limitaciones de uso	23
4 DESARROLLO	24
4.1 Introducción.	24
4.2 Herramientas empleadas.	25
4.2.1 Xcode.	25
4.2.2 SQLite.	25



4.2.3 Apple Developer Program.	26
4.2.4 Auto Layout.	26
4.2.5 Dropbox API.	27
4.2.6 Core Location.	28
4.2.7 iOS Charts.	29
4.2.8 iOS Simulator	29
4.3 Contenido del proyecto.	30
4.3.1 Modelo-vista-controlador	31
4.3.2 Storyboards.	32
4.4 Menús.	34
4.5 Launch Screen y App icon	35
4.6 Tipografía	36
4.7 Diseño y desarrollo del garaje.	37
4.7.1 Añadir coche	37
4.7.2 Mis coches.	39
4.7.3 Localización	40
4.7.4 Coche seleccionado	41
4.7.5 Manejo de fechas.	42
4.7.6 Esquema del apartado.	43
4.7.7 Descripción de archivos.	43
4.8 Diseño y desarrollo de historial.	44
4.8.1 Base de datos.	44
4.8.2 Edición del historial.	45
4.8.3 Estadísticas.	47
4.8.4 Esquema general del módulo.	48
4.8.5 Descripción de archivos.	48
4.9 Diseño y desarrollo de documentos.	48
4.9.1 Manejo de archivos	49
4.9.2 Sincronización de archivos.	50
4.9.3 Esquema general del módulo.	51
4.10 Diagrama de clases de la aplicación.	51
5 PRUEBAS Y RESULTADOS	53
5.1 Publicación en Apple App Store.	53
5.2 Estadísticas de la aplicación.	55
6 CONCLUSIONES Y TRABAJO	56
6.1 Conclusiones	56
6.2 Trabajo Futuro.	57
7 MANUAL DE USO DE LA APLICACIÓN	58
REFERENCIAS	65
PRESUPUESTO	66
PLIEGO DE CONDICIONES	67

## ÍNDICE DE FIGURAS

Figura 1.- Sony Ericsson GS84.....	4
Figura 2.- Comparativa iOS google play.....	5
Figura 3.- Aplicación Vehic.....	7
Figura 4.- Aplicación Mi coche.....	8
Figura 5.- Aplicación Car Controller.....	9
Figura 6.- Aplicación Ucar .....	10
Figura 7.- Diagrama de estados .....	14
Figura 8.- Desglose diario de uso de smartphones .....	17
Figura 9.- Cuota de mercado de plataformas móviles .....	18
Figura 10.- número de aplicaciones disponibles .....	18
Figura 11.- Comparativa yahoo weather iOS vs android .....	19
Figura 12.- Porcentaje de adopción iOS 8 Abril 2015 .....	20
Figura 13.- Pantallas de dispositivos apple .....	21
Figura 14.- Añadir librería SQLite Xcode .....	26
Figura 15.- Ejemplo de auto layout en iOS .....	27
Figura 16.- iOS simulator .....	29
Figura 17.- archivos en proyecto xcode .....	30
Figura 18.- Funcionamiento de la aplicación .....	31
Figura 19.- Modelo-vista-controlador .....	32
Figura 20.- Ejemplo storyboard .....	32
Figura 21.- Estilo de color .....	34
Figura 22a.- Menús de la aplicación .....	34
Figura 22b.- Menús de la aplicación .....	35
Figura 23.- Icono de la aplicación .....	35
Figura 24.- Launch screen .....	36
Figura 25.- sme-matriculas.es .....	38
Figura 26.- Icono no revisiones próximas .....	39
Figura 27.- Permiso localización .....	41
Figura 28.- Coche seleccionado .....	41
Figura 29.- Esquema de garaje .....	43
Figura 30.- Eliminar entrada del historial .....	46
Figura 31.- Alerta historial .....	46
Figura 32.- Introducir Swift en proyecto Objective-C .....	47
Figura 33.- Estadísticas .....	47
Figura 34.- Esquema Historial .....	48
Figura 35.- Ejemplo Collection View .....	49
Figura 36.- Manejo de la cámara .....	49
Figura 37.- Parámetros aplicación en Dropbox .....	50
Figura 38.- Esquema módulo documentos .....	51
Figura 39.- Diagrama de la aplicación .....	52
Figura 40.- Certificado de distribución .....	53
Figura 41.- Información de la aplicación .....	54
Figura 41.- App lista para vender .....	55

## ÍNDICE DE TABLAS

Tabla 1.- Escenas de XCode .....	33
Tabla 2.- base de datos SQLite de marcas y modelos de vehículo .....	37
Tabla 3.- Datos guardados del vehículo del usuario .....	38
Tabla 4.- Métodos de manejo de fechas .....	42
Tabla 5.- Tabla revisiones de SQLite .....	43
Tabla 6.- Archivos del módulo garaje .....	44
Tabla 7.- Tabla historial inicial .....	44
Tabla 8.- Tabla historial final .....	45
Tabla 9.- Archivos de historial .....	48
Tabla 10.- Métodos de integración con Dropbox .....	51

# 1 INTRODUCCIÓN

---

## 1.1 Motivación.

El mundo en el que vivimos está abarrotado de coches. Cada año crecen las cifras de ventas de vehículos, incluidos los últimos años de depresión económica. El automóvil es un medio de transporte que puede llegar a ser muy condicionante de la manera de vivir de las personas. Pero, ¿conocemos bien nuestro coche?, ¿sabemos llevar a cabo un correcto mantenimiento del mismo? La mayoría de los ciudadanos se limita a acudir al taller cuando se produce alguna avería. Es por esto que hemos desarrollado una aplicación que permita llevar a cabo un seguimiento más detallado de nuestro vehículo desde nuestro smartphone.

Nuestra aplicación está orientada a aportar información al usuario sobre el mundo del automóvil. El sector del motor produce, solamente en España, 2.6 millones de unidades anuales. El parque automovilístico español registró cerca de 27 millones de coches en 2014, siendo la media de edad de estos de 13 años. Estos datos nos hacen pensar en que la lista de posibles clientes que estén interesados en descargar la aplicación sea muy elevada, favoreciendo los futuros beneficios de la aplicación.

El objetivo principal de la aplicación es la de informar al usuario de las revisiones de distintos aspectos del vehículo según su kilometraje, estableciendo los tiempos en los que estas deben realizarse de una manera intuitiva y ordenada, añadiendo:

- Avisos al usuario de revisiones próximas.
- Consulta rápida de la información económica de los gastos del vehículo.
- Copia de la documentación necesaria del vehículo y posibilidad de subirla a la nube.
- Calculadora de fechas de matriculación.
- Uso de la localización para llevar a cabo un seguimiento más detallado de las distancias recorridas de los vehículos.

## 1.2 Objetivos.

La aplicación está orientada a aquellos que quieran iniciarse al conocimiento del mundo del automóvil de una manera fácil y ordenada, asimismo puede emplearse para guardar aquella información importante acerca del vehículo y del cliente de una manera segura. Esta aplicación se ejecutará sobre dispositivos Apple, en especial sobre los iPhone 5,6 y 6+, siendo compatible también con iPad con software iOS 7.0 o superior.

En línea con lo comentado anteriormente, el principal objetivo del PFC es ofrecer una aplicación que contenga toda la información de lo que acontece al vehículo, tanto de la información relativa a las próximas revisiones y sus precios como de todos aquellos gastos derivados del mismo (multas, lavados, seguros, etc). Toda esta información se almacenará en una base de datos que la aplicación manejará para presentar las estadísticas correspondientes.

Con la finalidad de atraer a un mayor número de usuarios, la aplicación ofrece otras herramientas interesantes. Junto con la anteriormente mencionada gestión de documentación del vehículo en la nube se encuentra la posibilidad del cálculo de la fecha de matriculación de cualquier vehículo, así como el uso de las herramientas de geolocalización de nuestro dispositivo móvil para contabilizar la distancia recorrida en un trayecto. Todas estas mejoras se introducen para incentivar el uso de la aplicación en aquellos clientes que posean un conocimiento avanzado de la gestión automovilística.

Adicionalmente contamos con diversas bases de datos creadas específicamente para el desarrollo de la aplicación. La primera cuenta con 600 vehículos de distintas marcas en las que el usuario podrá encontrar información sobre su vehículo, y la segunda consta de 1200 matrículas con las que la aplicación es capaz de realizar el cálculo de la fecha de matriculación de manera autosuficiente.

La aplicación esta orientada a dispositivos Apple y podrá ser descargada de manera gratuita en el App Store.

Desde el punto de vista educacional, el desarrollo se ha llevado a cabo con el objetivo de aprender un nuevo lenguaje de programación alejado parcialmente de los lenguajes C y

Java que se han asentado durante la carrera universitaria. Así como el empleo de bases de datos SQLite desde cero.

### **1.3 Metodología de trabajo.**

El plan de trabajo llevado a cabo para completar el proyecto puede compartimentarse en distintos puntos, que se detallan a continuación.

1. Estudio del estado del arte. Contiene un análisis de las principales aplicaciones centradas en el automóvil, observando sus posibilidades y debilidades.
2. Aprendizaje y recopilación del lenguaje de programación Objective-C y uso de bases de datos SQLite.
3. Análisis de la programación de la aplicación.
4. Testeo del correcto funcionamiento de la aplicación en distintos dispositivos.
5. Documentación.

### **1.4 Organización de la memoria.**

- Capítulo 1. Introducción.
- Capítulo 2. Estado del arte.
- Capítulo 3. Diseño.
- Capítulo 4. Desarrollo.
- Capítulo 5. Pruebas y resultados.
- Capítulo 6. Conclusiones y trabajo futuro.
- Capítulo 7. Manual de uso de la aplicación.

## 2 ESTADO DEL ARTE

---

Este punto se concentrará en analizar aquellas aplicaciones que ofrece el mercado de Apple centradas en la gestión mecánica de vehículos. Así como de los puntos fuertes y débiles del sistema operativo de la compañía de la manzana.

La finalidad del estudio es la de extraer toda información positiva y negativa de la situación de estas aplicaciones, que conlleven una mejora de las características del proyecto.

### 2.1 Introducción.

La idea principal del smartphone en su concepción fue la de unir dos elementos que ya se encontraban presentes de manera habitual en la vida cotidiana, dichos elementos eran una PDA (*Personal Digital Assistant*) y un teléfono móvil. El primer dispositivo que cumplió estos requisitos fue el IBM Simon, que surgió en 1992, además de sus importantes avances constaba de una pantalla enteramente táctil. Desafortunadamente, su peso excesivo de 500 gramos y su escaso rango de uso (solo funcionaba en 190 ciudades) marcaron su destino.

El primer dispositivo que empleó el término smartphone fue el Ericsson GS888 que fue el más avanzado de su época, el teléfono contaba con funciones de correo electrónico, navegación web, conexión a PC, etc.



FIGURA 1.- SONY ERICSSON GS88

El auge de los teléfonos inteligentes comenzó con la llegada del nuevo milenio, gracias a marcas como HTC, Palm OS y Windows Pocket PC, provocando en el 2007 la revolución del concepto de smartphone con la irrupción del iPhone de Apple [1].

## 2.2 Apple Inc.

Apple Inc. es una compañía norteamericana fundada por Steve Jobs, Steve Wozniak y Ronald Wayne el 1 de Abril de 1976. La compañía está centrada en el diseño, desarrollo y venta de electrónica para el consumidor, software, servicios online y ordenadores personales. Entre sus productos más conocidos se encuentran el iPod, iPhone, iPad y la línea de ordenadores Mac.

### 2.2.1 iPhone y App Store.

El primer iPhone fue presentado el 9 de enero de 2007 y su posterior venta comenzó el 29 de Junio de 2007, con un éxito sin precedentes. Este primer dispositivo aún no contaba con mercado de aplicaciones, aspecto fundamental en la situación actual de los smartphones. La introducción de la App Store no fue hasta un año después, abriendo un universo de posibilidades con aplicaciones y juegos. Desde entonces los terminales de Apple se renuevan cada año en los que hay mejoras constantes de procesador y herramientas software.

A día de hoy y pese a ser superada en número de aplicaciones por Google Play, los ingresos de la Apple App Store siguen duplicando a los de su más directo competidor, pese a que la cuota de mercado de Android es mucho mayor.

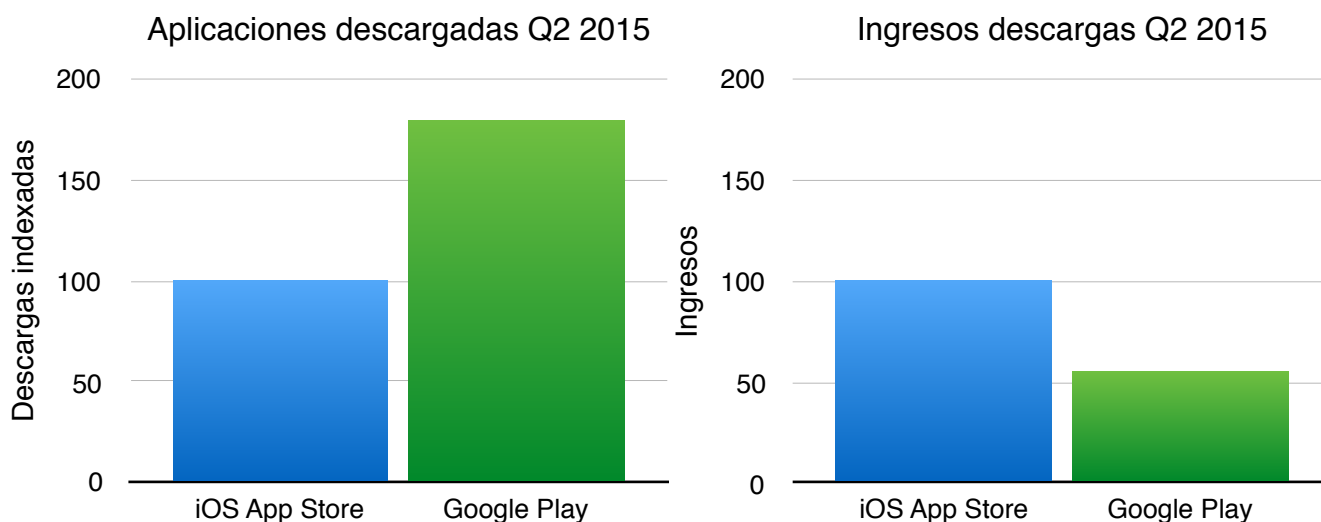


FIGURA 2.- COMPARATIVA IOS GOOGLE PLAY



Este es uno de los motivos por el que nos hemos decantado por el desarrollo en este sistema operativo, a pesar de llegar a un número menor de usuarios [2].

### **2.2.3 Objective-C.**

Objective-C es un lenguaje de programación orientado a objetos usado primordialmente en el desarrollo de software para OS X e iOS. Consiste en un superconjunto de C, lenguaje del que hereda su sintaxis, tipos primitivos y sentencias de control. Este lenguaje se empleaba de manera habitual en los ordenadores Mac de finales del siglo XX, como el sistema operativo de los iPhone estaba basado en estos ordenadores fue el lenguaje elegido para programar las aplicaciones nativas de iOS.

En la actualidad se esta tratando de sustituir este lenguaje por otro denominado Swift, que pretende una simplificación y mejora del código anterior. En estos momentos ambos lenguajes conviven de manera habitual en la mayoría de programas. La aplicación que hemos desarrollado se fundamenta prácticamente en su totalidad en Objective-C, con alguna inclusión esporádica de Swift por necesidad.

## **2.3 Análisis de aplicaciones similares.**

Existe una gran cantidad de aplicaciones que se centran en los automóviles, la mayoría de ellas centradas en la navegación y en la mejora de la experiencia del usuario en la conducción. A continuación se detalla un estudio de mercado en el cual se han analizado las aplicaciones mas características del sector.

### **2.3.1 Vehic.**



La aplicación Vehic desarrollada por Ignacio Garcia permite llevar a cabo el mantenimiento del vehículo. En ella se puede almacenar información relevante acerca de los cambios de aceite, reparaciones, consumos de combustible y otra información. Además permite la gestión de varios vehículos.

En cuanto a su funcionamiento, Vehic cuenta con un menú con la lista de los vehículos del usuario, pulsando en el vehículo correspondiente se accede a su información almacenada

La aplicación consta de cuatro puntos de información: Taller, Aceite, Neumáticos, Seguro e ITV. En dichos puntos la información solicitada es el precio y la fecha de los cambios producidos. En el caso del combustible permite hacer seguimientos de la media de gasto así como la de precio por litro. Como herramienta adicional hace empleo de la localización para guardar la posición en la que se ha aparcado el vehículo.

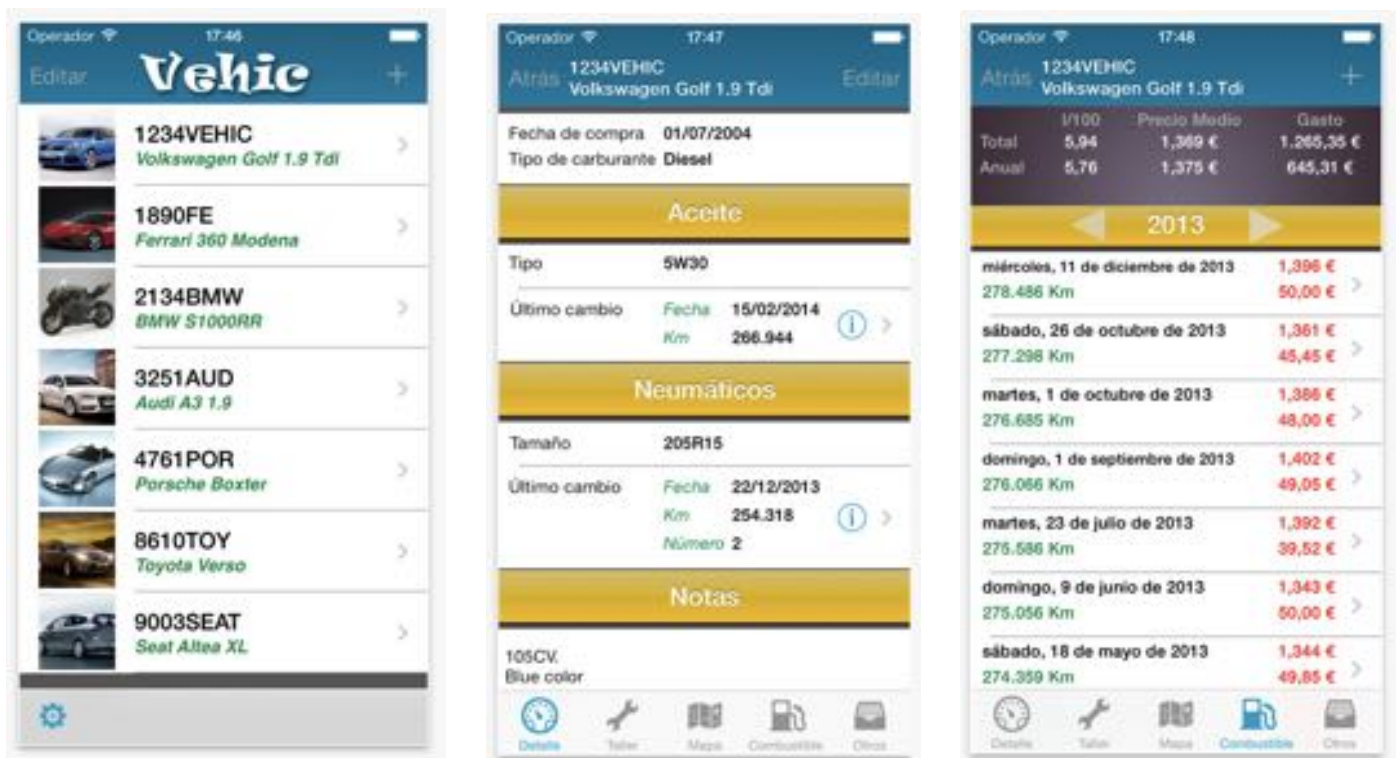


FIGURA 3.- APLICACIÓN VEHIC

Ventajas de la aplicación.

- + Interfaz intuitiva y simple.
- + Manejo estadístico estricto de los elementos del vehículo.
- + Posibilidad de manejo de varios automóviles.

Desventajas de la aplicación.

- Poco empleo de animaciones y herramientas visuales.
- Necesidad en algunos casos de introducir demasiados datos, como por ejemplo el precio de la gasolina por litro exacto.
- Uso lento de la aplicación.

La aplicación se encuentra disponible en la App Store por 1.99 €.

### 2.3.2 Mi Coche.



La aplicación Mi coche desarrollada por el equipo de Pipcode permite llevar a cabo un seguimiento de los gastos del vehículo. Además, la aplicación muestra multitud de estadísticas y la posibilidad de introducir las fechas de las revisiones pendientes.

Como añadido se ofrece la posibilidad de emplear el geolocalizador del smartphone en los trayectos, con el objetivo de que la base de datos se actualice de manera automática y obtenga estadísticas de los trayectos del usuario.

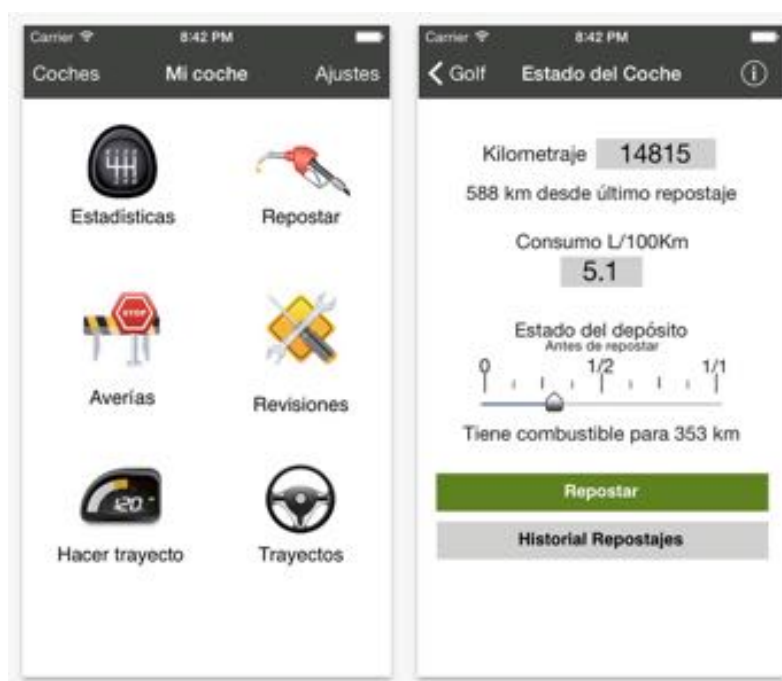


FIGURA 4.- APLICACIÓN MI COCHE

Ventajas de la aplicación.

- + Introducción rápida de datos.
- + Uso de geolocalización.
- + Introduce avisos de revisiones próximas.

Desventajas de la aplicación.

- Escaso atractivo visual.
- Mucha de la información que maneja ya es controlada por el ordenador del coche.
- Las revisiones próximas se deben introducir de forma manual.

La aplicación se encuentra disponible en la App Store por 1.99 €.

### 2.3.3 Car Controller.



La aplicación Car Controller desarrollada por Alberto Novo es, de igual manera que las anteriores, capaz de añadir nuevos vehículos y añadir a estos diversos datos. Los datos que maneja la aplicación son repostajes, averías, neumáticos y revisiones. La versión actual de la aplicación cuenta con una puntuación de 3.5 estrellas sobre 5.

Después de añadir los datos del vehículo del usuario el programa permite comenzar a introducir gastos del vehículo. La manera de añadir puntos de datos es mediante pestañas que aumentan o disminuyen la cantidad a fijar por el usuario. La aplicación es fácil e intuitiva y ofrece la posibilidad de añadir mucha información acerca de un gasto determinado.



FIGURA 5.- APLICACIÓN CAR CONTROLLER

Ventajas de la aplicación

- + Interfaz sencilla ordenada.
- + Gestión de una gran cantidad de datos.

Desventajas de la aplicación.

- No dispone de calculador de matrículas para obtener la fecha del vehículo.
- Su uso a pie de calle es lento por la gran cantidad de datos que necesita.
- No introduce estadísticas de los consumos realizados.
- Aplicación no optimizada para los nuevos terminales (iPhone 6 y 6+).

La aplicación se encuentra disponible de manera gratuita en la App Store.

### 2.3.4 uCar.



La aplicación uCar desarrollada por Uhuru Labs es una de las más conocidas dentro del sistema operativo android, cuenta con una valoración de 3,9 sobre 5 y acumula cerca de 50.000 descargas. las premisas de uCar son las mismas que se manejan en las anteriores aplicaciones, con la diferencia de que esta puede exportar e importar los datos en CSV y XML.

Su funcionamiento tampoco sale de las pautas de este tipo de aplicaciones, primeramente se añade un vehículo al garaje y después se maneja su información relacionada. Como añadidos se encuentra la localización de gasolineras y un acceso rápido para introducir gastos de repostaje de manera inmediata.



FIGURA 6.- APLICACIÓN UCAR

Ventajas de la aplicación.

- + El aspecto visual de la aplicación es moderno
- + El aspecto estadístico ofrece una gran cantidad de información de manera concisa.

Desventajas de la aplicación.

- No incluye avisos de revisiones pendientes.
- No introduce utilidades para ofrecer información al usuario inexperto.

La aplicación se encuentra disponible en Google Play sin coste alguno.

### **2.3.5 Causas comunes en la eliminación de aplicaciones.**

Un punto importante a la hora de contemplar el estado actual de las aplicaciones móviles es prestar especial atención a los principales motivos por los que el usuario desinstala una aplicación. Existen estudios que se dedican especialmente a seguir las impresiones negativas más comunes que los clientes no soportan y que se citan a continuación

- Exceso de notificaciones.

Las notificaciones son una herramienta que puede llegar a ser de gran utilidad y que se emplean en muchas aplicaciones. Pero usadas sin control son irritantes para el usuario.

- Proceso de registro complejo.

Algunas aplicaciones están vinculadas a un servicio web que permite sincronizar y otras opciones. El problema es que completarlo resulta un proceso engorroso y lento.

- Tiempos de carga excesivo.

El mercado de la tecnología ha evolucionado mucho y los tiempos de carga tradicionalmente lentos en la historia de los dispositivos móviles han llegado a su fin. El usuario desea acceder de manera instantánea a la información.

- Mala interfaz.

El aspecto visual es uno de los aspectos más importantes en una aplicación. Es importante que los programadores tengan conceptos de diseño que les ayude para convertir un buen proyecto en uno brillante.

- Publicidad intrusiva.

Hay aplicaciones que provocan en el usuario confusión entre los anuncios y los elementos de la propia aplicación. Esto resulta muy molesto y es un aspecto negativo a considerar.

- Iniciar sesión en las redes sociales.

Existen aplicaciones que obligan a registrarnos mediante el perfil de Facebook, Twitter, etc. El motivo de este registro es el de publicar información en tu nombre a modo de spam.

### **2.3.6 Conclusiones.**

Después del repaso de los sistemas operativos y de las aplicaciones disponibles de temática similar a la que queremos desarrollar, la decisión de implementar nuestro proyecto en iOS se fundamenta en la gran cantidad de ingresos que se generan la App Store en comparación con Google Play, siendo la cuota de mercado de este último mayor. Es, por tanto, el entorno más apropiado en este sentido.

Adicionalmente, después de realizar un estudio de mercado de ambos sistemas operativos, se puede observar que el número de aplicaciones existentes en iOS parecidas a la que queremos desarrollar es mucho menor, siendo además en su mayor parte anticuadas y con un éxito escaso. Esta situación se debe a que en su mayoría requieren de un pago de alrededor de 2 €, situación que una gran parte de los usuarios no contempla. Recientes estudios sobre la intención del consumidor a la hora de asumir gastos en cualquier mercado de aplicaciones revelan que una cantidad significativa de ellos no contempla realizar pagos de más de 0.99 € por aplicación.

En cuanto a la calidad y herramientas que nos ofrecen las aplicaciones ya disponibles en el mercado, el mecanismo de funcionamiento es semejante en su mayor parte. Las herramientas adicionales son las que marcan la diferencia entre unas y otras. Existen varios aspectos a tener en cuenta a la hora de gestionar vehículos en el smartphone.

#### **1. Rapidez de uso**

En situaciones de la vida cotidiana en las que el uso del tiempo puede ser limitado, como puede ser un repostaje, es importante que los datos puedan introducirse de una manera ágil e intuitiva para el usuario.

#### **2. Presentación de estadísticas.**

Los datos deben presentarse de una manera organizada y ofrecer distintas maneras de visualización de los gastos que el usuario ha realizado.

#### **3. Avisos de revisiones pendientes.**

Para el usuario inexperto en el tema le puede resultar de poca ayuda la gestión manual de las revisiones de su vehículo. De modo que es importante que se añada información y ayuda en este sentido.

## 3 DISEÑO

---

Seguidamente se realiza un análisis exhaustivo del proyecto, de las principales propiedades y funciones del mismo. Adicionalmente se comentarán los requisitos, puntos fuertes y limitaciones de la aplicación y se justificarán las decisiones tomadas durante el diseño del proyecto.

### 3.1 Objeto del proyecto.

Como se ha explicado anteriormente, uno de los principales objetivos tenidos en cuenta a la hora de planificar el proyecto ha sido el de acercar la mecánica automovilística al conductor cotidiano.

La herramienta más útil por su gran avance en los últimos años para llevar a cabo el acercamiento entre el usuario y su vehículo es su dispositivo móvil. Actualmente la cantidad de aplicaciones de temática automovilística es mayor que nunca y estas abarcan una gran cantidad de aspectos relacionados (navegación, gestión, taxi, aparcamiento, etc).

Mientras que muchos aspectos de la conducción disponen de numerosas aplicaciones con años de desarrollo, aquellas destinadas a la gestión de vehículos han permanecido en un segundo plano. Después de analizar las aplicaciones que ofrecen utilidades similares en el apartado 2 ESTADO DEL ARTE concluimos que las grandes carencias que presentaban se podían reducir a dos: su alto precio (teniendo en cuenta el precio medio de gasto por aplicación) y su escaso atractivo visual.

Teniendo en cuenta todos estos aspectos, hemos desarrollado una aplicación fundamentada en el principio de rapidez y el de una interfaz sencilla e intuitiva en la que ha predominado el aspecto visual, presentando animaciones y aportando una interacción fluida con el usuario.

La aplicación muestra además otros aspectos para expandir su área de trabajo fuera de la gestión de vehículos, ofreciendo un espacio donde guardar una copia de la documentación del usuario y su vehículo, herramientas de cálculo de fechas de matriculación y posibilidad de vinculación con el servicio de alojamiento en la nube Dropbox.



### 3.2 Diagrama de estados.

En este apartado se mostrará una visión general del funcionamiento de la aplicación mediante un diagrama de estados en el que se detalla de manera simple el funcionamiento de la aplicación.

La sencillez es un aspecto importante en el diseño de cualquier programa, en los dispositivos móviles una solución para aprovechar la pantalla de una manera más eficiente es mediante el menú lateral. Consiste en un panel que se expande y contrae desde el lado izquierdo de la pantalla y muestra las opciones de navegación, resultando más visual y personalizable que el clásico menú principal.

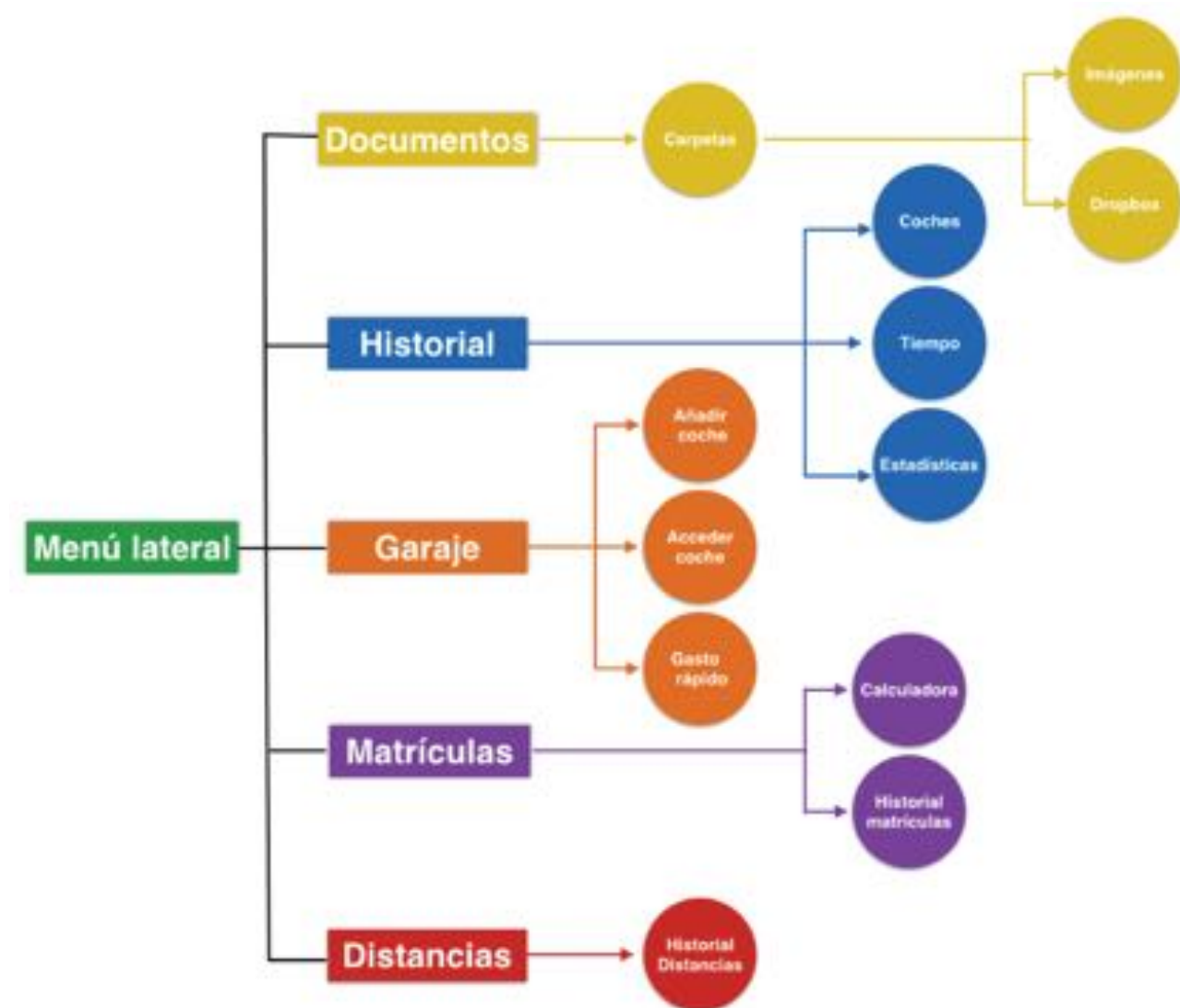


FIGURA 7.- DIAGRAMA DE ESTADOS

Como puede observarse en la figura anterior, el menú lateral se convierte en el enlace principal de las funciones de la aplicación. Sin embargo, la ventana de inicio por defecto corresponde al garaje, desde donde se añaden vehículos y se accede a su información.

La implementación detallada completa del modo de acceso a todas las funciones se describirá a continuación. La decisión del diagrama de estados en forma de árbol responde a la exigencia generalizada de simplicidad en cualquier aplicación. Además, el empleo del menú lateral proporciona una sensación de continuidad en la experiencia del usuario, motivo por el que se valora de forma positiva y un gran número de aplicaciones lo emplean.

### **3.3 Análisis de la aplicación.**

Las funciones completas que ofrece la aplicación al usuario se dividen fundamentalmente en seis partes: Garaje, historial, matrículas, documentos, usuario y distancias.

#### **3.3.1 Garaje.**

La primera función a destacar es el garaje, ya que corresponde al núcleo de la aplicación. Su primera función es la de añadir los vehículos deseados por el usuario, una vez rellenada la información solicitada referente al kilometraje y la matrícula, la aplicación establecerá la distancia media recorrida al mes y establecerá las fechas de las próximas revisiones. En caso de vehículos nuevos o medias incorrectas, esta podrá ser modificada más adelante por el usuario.

Una vez los vehículos se han introducido, esta ventana será la encargada de avisar al usuario cuando se aproxima una revisión en un periodo menor de tres meses. Ofreciendo la posibilidad de acceder a una nueva ventana con información detallada del contenido de las revisiones e información detallada de los gastos del vehículo, desde donde se completarán las revisiones y añadirán nuevos gastos.

Por ser la ventana de apertura de la aplicación por defecto se han incluido diversas herramientas con el fin de mejorar el tiempo de introducción de nuevos datos. Incluye 3 botones de acceso rápido: estadísticas, historial y gasto rápido. Las dos primeras corresponden a las mismas interacciones que ofrece el menú lateral mientras la tercera ofrece una vista donde se pueden introducir rápidamente el nuevo gasto que ha producido el usuario acerca de un vehículo.

Por último se encuentra una barra de localización que el usuario activará antes de realizar un trayecto y que almacenará la distancia recorrida por el vehículo.

El empleo de la localización en esta aplicación requiere un consumo de batería por debajo de la media (5 % por cada hora de uso) y el dispositivo móvil no precisa de estar desbloqueado. Una vez se alcance el final del mes una nueva ventana preguntará si se desea registrar la distancia acumulada mediante localización en el vehículo, en caso de que el usuario no lo desee continuará registrando la media mensual.

### **3.3.2 Historial.**

Apartado donde se encuentra toda la información de los distintos gastos que se han producido para cada vehículo, ordenado de más reciente a más antiguo. La ventana ofrece la posibilidad de desglosar esa información en meses ordenados cronológicamente

Se añade la posibilidad de contemplar una gráfica de barras dividida cronológicamente donde se encontrará la información del gasto mensual total y el correspondiente a cada vehículo. Las entradas de datos de cada gasto podrán ser eliminadas en caso de ser incorrectas.

### **3.3.3 Usuario.**

Espacio donde se podrá guardar información relativa al usuario de la aplicación (fotografía, nombre, apellidos) y se encuentra la función de vincular la cuenta de Dropbox, destinada específicamente a la gestión de documentos del usuario y que se detallará en su correspondiente apartado.

### **3.3.4 Documentos.**

Esta funcionalidad ofrece la posibilidad de almacenar una copia de aquella documentación relativa al vehículo que el usuario desee, incluyendo varios apartados que agrupan la información. La manera de introducir los archivos se producirá mediante fotografías, acceso a la librería del dispositivo móvil y descarga del sistema de almacenamiento en la nube Dropbox. Este último también proporcionará la función de subida de los datos almacenados.

### 3.3.5 Matrículas.

La quinta herramienta proporcionada por la aplicación se basa en un calculador de matrículas en el que puede obtener la fecha de matriculación de cualquier vehículo, mostrando en la misma ventana las últimas matrículas que se han calculado por el usuario.

### 3.3.6 Distancias.

Tabla que muestra los datos que se han obtenido mediante el uso de la localización de las distancias de los trayectos del usuario. Esta interfaz permite también el borrado de cualquier punto de datos guardado.

## 3.4 Entorno de la aplicación.

### 3.4.1 Sistema operativo.

El sistema operativo que se ha elegido para el desarrollo del proyecto ha sido iOS. Los motivos que se han tenido en cuenta a la hora de incluir esa decisión se detallan a continuación.

El número de dispositivos Android ha superado hace ya tiempo a los de iOS tanto a nivel internacional como nacional. Pero todavía existen motivos por los que este entorno permanece atractivo para muchos desarrolladores. El primero de estas razones residen en la actividad de los usuarios con los propios dispositivos, los usuarios de Apple tienden a navegar más por internet y a hacer más compras con ellos



FIGURA 8.- DESGLOSE DIARIO DE USO DE SMARTPHONES

Como muestra la figura los usuarios de iPhone dedican 26 minutos de media más a atender a sus teléfonos. Además usando el teléfono en menor proporción.

Por tanto, los usuarios de la manzana emplean más tiempo en la búsqueda de nuevas aplicaciones. Unido a eso recientes estudios aseguran que son más fieles a las aplicaciones que se descargan, superando a Android en número de aplicaciones que se usan más de 10 veces en un 52 %.

Todos estos datos se traducen en las cifras oficiales de descargas en ambas plataformas de aplicaciones (figura 2.2.1). En ellas se puede observar que pese que Google Play dobla el número de descargas de aplicaciones el número de usuarios de Android es casi 6 veces mayor al de iOS, como puede observarse en la tabla siguiente.

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
2015Q2	82.8%	13.9%	2.6%	0.3%	0.4%
2014Q2	84.8%	11.6%	2.5%	0.5%	0.7%
2013Q2	79.8%	12.9%	3.4%	2.8%	1.2%
2012Q2	69.3%	16.6%	3.1%	4.9%	6.1%

Source: IDC, Aug 2015

FIGURA 9.- CUOTA DE MERCADO DE PLATAFORMAS MÓVILES

Junto con este dato, si atendemos al número de aplicaciones de ambos mercados de aplicaciones es claro que Apple App Store es la plataforma más atractiva históricamente para desarrolladores.

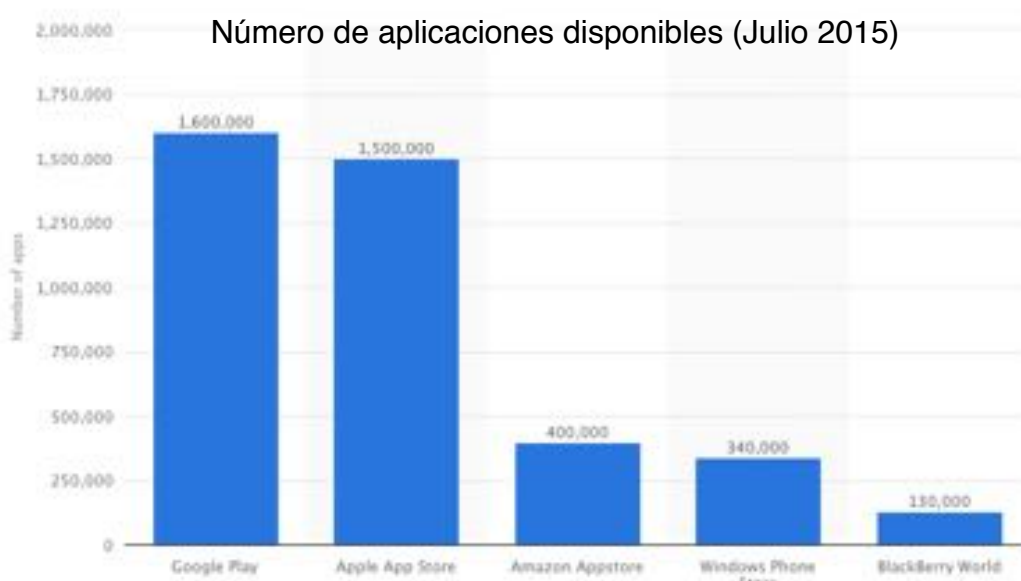


FIGURA 10.- NÚMERO DE APLICACIONES DISPONIBLES

Con un mercado de usuarios 6 veces menor, las aplicaciones disponibles en Apple App Store únicamente se encuentran 100.000 por debajo de Google Play.

Con relación a los ingresos de derivados de las aplicaciones, como muestra la figura 2.2.1 la plataforma de Apple duplica aún a la de Google. Aún así, la diferencia de ingresos se ha reducido considerablemente en los últimos dos años (2013 y 2014).

- El Factor estratégico.

Cualquier desarrollador de aplicaciones debe prestar especial atención en la tendencia de éxito que han tenido las aplicaciones más presentes en los terminales móviles.



FIGURA 11.- COMPARATIVA YAHOO WEATHER iOS VS ANDROID

Aplicaciones como Instagram, Clipboard, Flickr, y Vine comenzaron su desarrollo en la plataforma de Apple, donde después del éxito obtenido migraron hacia nuevas plataformas. Incluso plataformas propias de Google fueron implantadas primeramente en este mercado, como es el caso de Google Maps.

Según Evan Doll, cofundador de Flipboard, “iOS y Android, cada uno tiene diferentes fortalezas que se prestan a cosas diferentes. Los dispositivos con iOS son más predecibles en términos de tamaño de pantalla y capacidades, lo que es muy útil cuando estás desarrollando una nueva funcionalidad.”

Por todo lo anterior se decidió establecer el desarrollo del proyecto en iOS. Pese al auge de Google Play, iOS sigue ofreciendo una plataforma más atractiva para desarrolladores en cuanto a posibilidades de expansión de las aplicaciones. Sumado a la mejor disposición de sus usuarios a descargar y usar nuevas aplicaciones de pago en los terminales. No cabe duda de que Android seguirá madurando como plataforma y en algún

momento de los próximos años más compañías opten por el desarrollo de aplicaciones en primera instancia en Google Play. Hasta entonces, iOS es la apuesta más segura [4].

### 3.4.2 Requisitos del sistema operativo.

La propia política de Apple de no ofrecer actualizaciones de software a terminales antiguos limita la posibilidad de desarrollar las aplicaciones en versiones anteriores al sistema iOS 7. Además, el lanzamiento de iOS 8 en septiembre de 2014 estuvo salpicado por problemas y contratiempos que resintieron su porcentaje de adopción en los usuarios. Los principales obstáculos del nuevo sistema eran la gran cantidad de cambios internos e infinidad de nuevas herramientas para desarrolladores, que incluían incluso un cambio de lenguaje de programación.

Pese a estos problemas, en el primer trimestre de 2015 el nuevo sistema operativo se encuentra en plena madurez, llegando al 81 % de dispositivos.



FIGURA 12.- PORCENTAJE DE ADOPCIÓN iOS 8 ABRIL 2015

Atendiendo a esta figura, la decisión de las versiones del sistema en las que se iba a desarrollar la aplicación se concretó en iOS 7 e iOS 8, que suponen el 98 % de los dispositivos actuales. A medida que se introduzcan nuevas versiones del sistema operativo se introducirán los cambios oportunos para dar soporte a la aplicación.

### 3.4.3 El problema de las pantallas.

La introducción del iPhone 6 en septiembre de 2014 supuso una alegría para los aficionados a la marca, pero no tanto para los desarrolladores. Hasta entonces solamente se contemplaban dos resoluciones fijas que simplificaban el desarrollo, pero la introducción de nuevos tamaños introdujo el debate de la necesidad de crear experiencias diferentes para cada dispositivo.

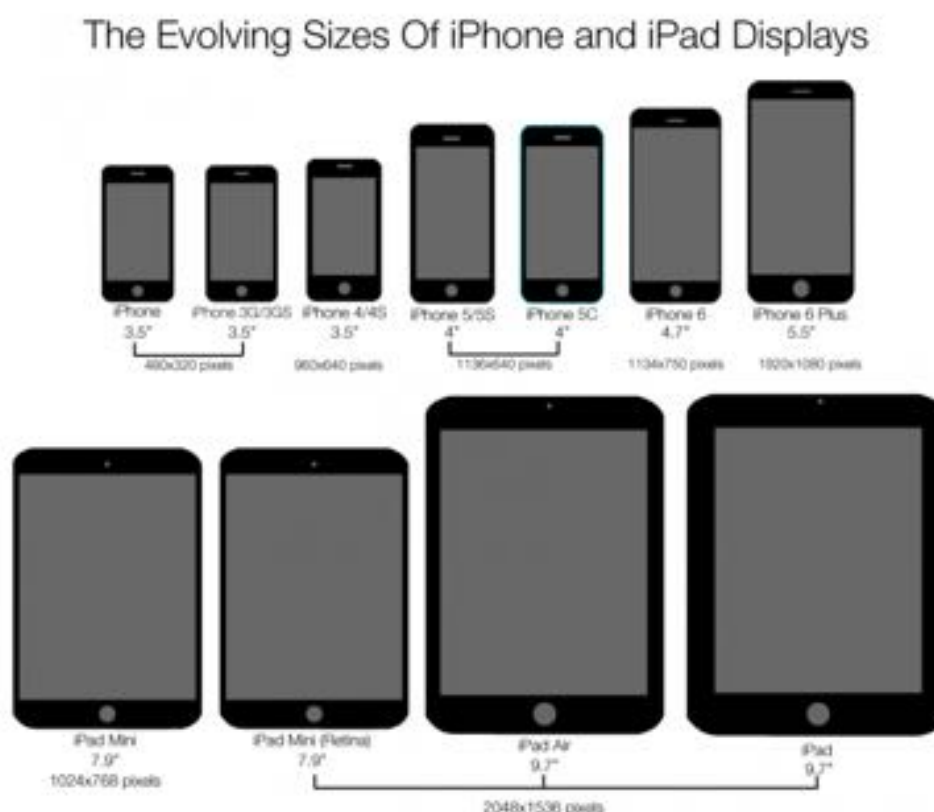


FIGURA 13.- PANTALLAS DE DISPOSITIVOS APPLE

La solución proporcionada por Apple consiste en la herramienta Auto Layout, que se describirá con detalle en la sección de desarrollo. Según esta herramienta los elementos de la interfaz creada por el programador se distribuirán en el espacio siguiendo unas pautas establecidas que también proporciona el desarrollador.

La interfaz empleada en el uso de Auto Layout en nuestro proyecto es la de iPhone en modo retrato (vertical). Esta decisión se tomó debido a la gran cantidad de pantallas en las que esta interfaz es compatible. Los elementos de la aplicación se distribuyen en cada pantalla de una manera apropiada para el caso de los smartphones, mientras que en las tabletas la aplicación se mostrará ampliada. Dependiendo del éxito del proyecto en los smartphones (en los que se focaliza este desarrollo) se estudiará la introducción de una interfaz propia para todos los modelos del iPad.



### **3.4.4 Requisitos de la aplicación.**

Sistema Operativo	iOS
Versiones	iOS 7.0 ó superior
Dispositivos	iPhone 5 iPhone 5S iPhone 6 iPhone 6+ iPhone 6s iPhone 6s+ iPad 2 iPad retina iPad Air iPad Air 2

## **3.5 Limitaciones**

### **3.5.1 Limitaciones de lenguaje.**

El desarrollo de la aplicación requiere la creación de una base de datos que contenga una guía de las fechas de matriculación de los últimos quince años, como la base de la aplicación se encuentra en España se decidió emplear la información relativa a este país. Al ser una aplicación centrada en el parque automovilístico español consideramos que la cuota de mercado hace ilógica la opción de ofrecer la aplicación en varios idiomas.

Una vez lanzada al mercado, y en vista de los resultados obtenidos en el número de descargas, se analizará la posible expansión de la aplicación en otros países, añadiendo a su vez nuevas bases de datos con la información relativa a estos.

### **3.5.2 Limitaciones técnicas**

La aplicación únicamente considera vehículos con el estándar de matriculación implantado a partir del año 2000, que permanece en la actualidad. Por tanto, para los vehículos anteriores a esta implantación la aplicación reducirá sus funcionalidades.

Los motivos de esta decisión se fundamentan en recientes estudios del parque de vehículos nacional. El porcentaje de coches con más de quince años, y por tanto con el anterior estándar de matrícula, no llega al 15% del total. Además, con la reciente mejora de la situación económica el número de vehículos nuevos vendidos los últimos meses ha

supuesto un aumento del 20 % con respecto a los datos de 2014. Lo que supone una reducción en el porcentaje de vehículos con matriculación antigua hasta alcanzar una previsión del 12% en 2016. Por tanto consideramos que esta limitación no será un inconveniente en el número de clientes interesados en la aplicación.

Por otra parte, los dispositivos electrónicos de Apple establecen un porcentaje de memoria que puede usar la aplicación durante su funcionamiento según la que disponga el terminal empleado. Esto podría suponer un problema en la gestión de documentos que posee la aplicación. Sin embargo, a efectos prácticos supondría problemas de memoria para casos en los que se encuentren cerca de 200 archivos. El número de archivos muy difícilmente alcanzará esa dimensión, por tanto tampoco será un inconveniente.

### **3.5.3 Limitaciones de uso**

- La aplicación no contempla cambios de rotación en la pantalla.
- La introducción de gastos relativos a cambios de neumáticos no es reversible.
- El error en el la distancia de trayecto calculada por geolocalización puede llegar hasta 1 kilómetro.
- Como se ha comentado en el apartado 3.5.1, limitación al mercado español.

Las limitaciones presentadas serán solventadas en el futuro por actualizaciones de versión y mediante feedback de los propios usuarios, como se detallará en el apartado 6 de la memoria.

## 4 DESARROLLO

---

En el siguiente punto se incluirá un análisis de la parte técnica del desarrollo del proyecto. Se explicará el modo en el que se ha programado las distintas partes de la aplicación, las decisiones tomadas frente a los distintos problemas que surgieron durante su desarrollo y las relaciones existentes entre sus distintas partes.

### 4.1 Introducción.

Antes de comenzar el desarrollo y debido a la falta de experiencia tanto con el lenguaje de programación como con el entorno de desarrollo, se consultaron diversos cursos en los que se proporcionaba información sobre programación. A continuación se consultó información con técnicos de concesionarios sobre que conceptos podrían introducirse en la aplicación.

Los puntos mas importantes en los primeros pasos fueron:

- Curso online “Developing iOS 7 apps for iPhone and iPad”. Universidad de Standford.

En este curso se introducían desde los conceptos fundamentales en el desarrollo de aplicaciones en Objective-C hasta los aspectos más avanzados, incluyendo ejercicios para realizar por el alumno en cada tema.

- Lectura del libro “Beginning iOS 8 programming with swift”. AppCoda.

Este libro contiene información acerca del nuevo lenguaje de programación Swift, explicado de una manera amena con ilustraciones y capturas de pantalla con los temas que se tratan en él. Además incluye todo el código fuente empleado e interfaces de prueba e iconos para realizar la aplicación.

- Consulta en distintas webs y foros especializados.

Se consultaron varias páginas web con información interesante sobre tutoriales y problemas que suele encontrar el programador a la hora de desarrollar la aplicación. principalmente se consultaron las páginas AppCoda, StackOverflow y GitHub.

- Contratación del paquete gráfico “iOS designer bundle”. Cult of Mac Deals

Primer acercamiento al mundo del diseño gráfico, este paquete contenía diversas pautas a la hora de realizar aplicaciones para iOS y una gran cantidad de vectores e iconos para realizar el proyecto.

## 4.2 Herramientas empleadas.

### 4.2.1 Xcode.



Publicado por primera vez en Octubre de 2003 junto a la versión 10.3 de Mac OS X, Xcode es el entorno de desarrollo integrado creado por apple que incluye todas las herramientas necesarias para desarrollar aplicaciones y programas.

El programa incluye herramientas gráficas para crear interfaces de usuario a la vez que maneja distintos lenguajes de programación, puede compilar código C, C++, Objective-C, Objective-C++, Applescript, java y Swift. La última versión (Xcode 6) cuenta con un diseño de interfaz que unifica la codificación, pruebas y depuración dentro de una única ventana.

Xcode está disponible de forma gratuita y se puede descargar desde la Mac App Store [5].

### 4.2.2 SQLite.

La aplicación que vamos a desarrollar se fundamenta principalmente en la gestión de datos, por tanto un aspecto muy importante de la misma es el manejo de bases de datos de manera rápida y eficiente de los recursos del dispositivo, y la forma más sencilla de conseguirlo es mediante el empleo de SQLite.

SQLite es un motor de bases de datos basado en SQL, su código es de dominio publico y es la herramienta de gestión de datos con mayor presencia en aplicaciones alrededor del mundo. La mayor ventaja que presenta SQLite frente a sus competidores es la ausencia de un servidor de procesos separado, proporcionado múltiples tablas, vistas e índices en un mismo archivo.

Existen otras librerías con mayores herramientas y posibilidades que SQLite. La decisión de emplear esta base de datos es su escaso tamaño y su reducido tiempo de acceso.

Para el empleo de la base de datos en Xcode es necesario añadir la librería correspondiente, como se muestra a continuación [6].

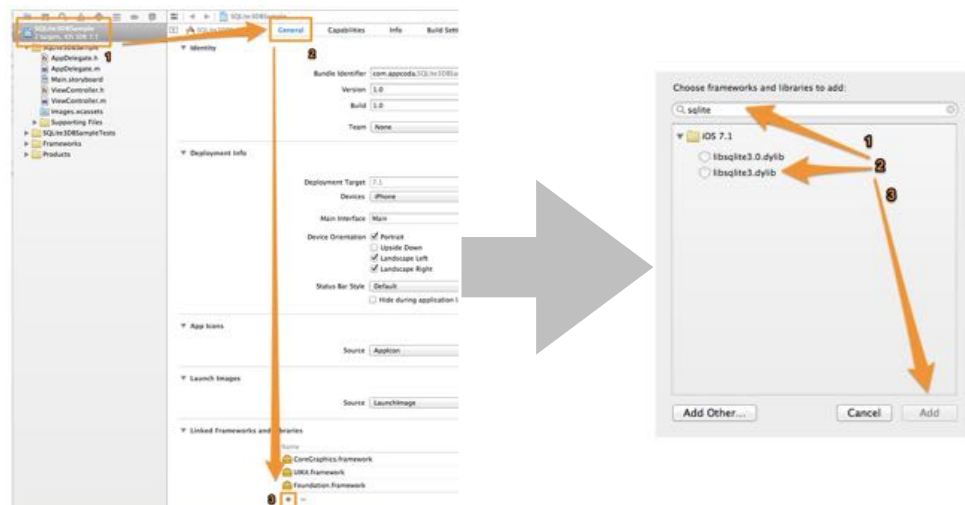


FIGURA 14.- AÑADIR LIBRERÍA SQLITE XCODE

#### 4.2.3 Apple Developer Program.

Un requisito necesario a la hora de publicar nuestro proyecto en el mercado de aplicaciones de iOS es necesario obtener una cuenta de desarrollador. Las ventajas de este programa incluyen betas del nuevo software y herramientas de test para analizar la gestión de recursos y posibles fallos de la aplicación. El precio de la cuenta es de 99 \$ al año y el 30% de los ingresos de cada aplicación que se encuentre en el mercado de aplicaciones.

A pesar de su elevado precio el programa de desarrolladores ofrece descuentos para estudiantes y universitarios. No cabe duda que este es un punto débil en el desarrollo en esta plataforma pero ofrece diversas facilidades de empleo de las distintas capacidades del dispositivo y proporciona un mecanismo de distribución propio de la aplicación.

#### 4.2.4 Auto Layout.

La aplicación está destinada a diferentes dispositivos, que se traduce en diferentes interfaces acordes al tamaño de pantalla del usuario (iPhone 5, iPhone 5S, iPhone 6, iPhone 6+, etc). La manera de simplificar la creación de varios proyectos con distintos tamaños de pantalla se realiza mediante Auto Layout.

Esta herramienta consiste en la creación de una interfaz en la que los elementos que figuran en ella se distribuyen en la pantalla de manera adaptativa, respondiendo a cambios de pantalla y de orientación del smartphone. Permite al desarrollador crear interfaces de usuario con alto grado de personalización en varios dispositivos al mismo tiempo.

Su mecanismo de funcionamiento se basa en el empleo de restricciones (constraints). Cada elemento contiene sus propias restricciones que establecen las líneas generales del comportamiento visual del elemento como el tamaño, relación de aspecto, posición dentro de la pantalla, etc. El compilador ofrece información sobre si las restricciones establecidas son suficientes para un correcto funcionamiento mediante un aviso en el programa.



FIGURA 15.- EJEMPLO DE AUTO LAYOUT EN IOS

Auto Layout ha recibido diversas críticas de antiguos desarrolladores por su dificultad de empleo, pero se decidió su introducción en el desarrollo del proyecto debido a las facilidades que propone en la difusión de la aplicación en distintos dispositivos, punto primordial en nuestro objetivo de lograr la mayor penetración en el mercado posible [7].

#### 4.2.5 Dropbox API.

Dropbox lanzó su API para desarrolladores en Octubre de 2011 para crear aplicaciones basadas en su servicio de almacenamiento en la nube, estas deben ser registradas en Dropbox donde se les proporcionará una clave única que se incorpora al proyecto y que permitirá la vinculación de la cuenta del usuario. La librería proporciona un manejo total de

todas las funciones del servicio, tanto de descarga de archivos como de subida de estos. Así como la eliminación de archivos y modificación de datos del usuario.

Esta utilidad permite acceder a la cuenta de dos maneras: mediante acceso completo al directorio del usuario o a través de la creación de una carpeta dedicada. La decisión del modo de acceso para nuestra aplicación se basó en la necesidad de agrupar la información que vamos a manejar (esencialmente documentos), por lo que se optó por la carpeta dedicada a la aplicación.

Dropbox es uno de los sistemas más extendidos entre los usuarios, con presencia en una gran cantidad de aplicaciones y con un tamaño básico suficiente para manejar los documentos y fotografías necesarias en nuestra aplicación. Estos motivos son los que han decantado el uso de este servicio con respecto de sus competidores (Google Drive ó iCloud) [8].

#### **4.2.6 Core Location.**

Usar información basada en la localización es una herramienta muy útil para mantener al usuario conectado con el mundo que le rodea y puede mejorar la experiencia global del usuario con la aplicación. Existen dos grandes grupos de información de localización en Core Location: servicios de localización y mapas. La aplicación se centrará en los primeros ya que, al ser una aplicación con un propósito gestor, no ofrece servicios de navegación.

La introducción de este servicio en nuestro proyecto obedece a que el cálculo de kilómetros recorridos al mes en muchos casos puede no corresponder a la realidad. De manera que mediante los datos proporcionados por este método permitirá al usuario mantener un control más estricto en la información del vehículo.

De igual manera, uno de los aspectos más importantes cuando se maneja información sobre localización es el alto consumo de batería. Core Location ofrece distintos rangos de cambios en la localización con mayor error de precisión pero que aportan un consumo mucho menor. En nuestro proyecto se consideran únicamente cambios significativos de posición, decisión razonable atendiendo a que la aplicación se orienta a vehículos en los que no se precisa de un motor de navegación.

#### 4.2.7 iOS Charts.

La API de iOS Charts es una librería introducida en 2014 por Daniel Cohen Gindi como una versión para la plataforma de Apple de la popular librería MPAndroidChart creada por Philipp Jahoda. Dentro de esta librería se pueden encontrar varios tipos de gráficas que se pueden emplear en cualquier aplicación de manera sencilla.

El uso de gráficas contribuye al usuario a comprender mejor la información y ayuda a organizar la información. La librería aporta adicionalmente reconocimiento de gestos en las gráficas lo que contribuye a una mejora en la experiencia del usuario.

La manera de introducir esta librería en el proyecto mantiene la misma pauta que SQLite (figura 3.4.29) con la salvedad de que se ha introducido manualmente desde la web de GitHub correspondiente a Daniel Cohen [9].

#### 4.2.8 iOS Simulator

El simulador integrado en el programa Xcode permite obtener rápidamente prototipos y comportamientos de la aplicación durante el proceso de desarrollo. iOS Simulator funciona sobre Mac y se comporta como una aplicación estándar de esta mientras simula el comportamiento de un iPhone, iPad o Apple Watch. Se trata de una herramienta interesante para un testeo preliminar del comportamiento de la aplicación desarrollada.

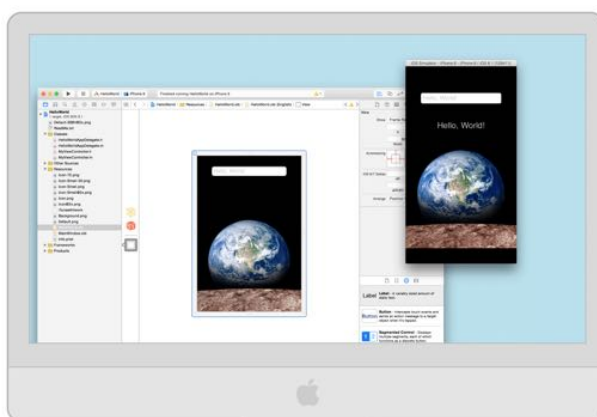


FIGURA 16.- IOS SIMULATOR

Simular el proyecto mediante esta herramienta permite al desarrollador encontrar problemas graves en el código, uso de herramientas especiales (como simulación de carrera a pie, viaje en coche, viaje en bicicleta, etc), y permite acceder al entorno de iOS antes de pertenecer a un plan de pago de programador de Apple.



### 4.3 Contenido del proyecto.

En esta sección se introducen los conceptos más importantes en cuanto a el tipo de archivos que se encuentran en todos los proyectos de XCode, junto con una explicación de su funcionamiento en general.

Cuando se crea un proyecto en blanco en el programa se incluyen los siguientes archivos.

AppDelegate.h	ViewController.h	Main.Storyboard
AppDelegate.m	ViewController.m	

Como Objective-C es un superconjunto de C mantiene su estructura de archivos típica, con el archivo de cabecera .h y el archivo de implementación .m. Sin embargo, el lenguaje de programación está orientado a objetos, de modo que cuando programamos aplicaciones iOS creamos y definimos clases que representan distintas entidades, que posteriormente se usan para crear instancias llamadas objetos. Por tanto, el proyecto de XCode se fragmenta como describe la siguiente figura.



FIGURA 17.- ARCHIVOS EN PROYECTO XCODE

El archivo ViewController mostrado en la figura 4.3 se encarga, como su nombre indica, de la gestión y control de una ventana. De modo que el storyboard contiene la ventana que maneja ViewController, además se encarga del gestionar del flujo de ventanas que forman la aplicación.

El archivo AppDelegate forma el punto de entrada de la aplicación, es el responsable de crear un objeto de tipo ViewController cuando recibe la información de que la aplicación ha sido lanzada.

El objeto ViewController es responsable de preparar la ventana que se describe en el storyboard y de enseñarla en la pantalla para que el usuario interactúe con ella. Seguidamente se incluye un diagrama en el que se muestra el comportamiento de todo proyecto en XCode.

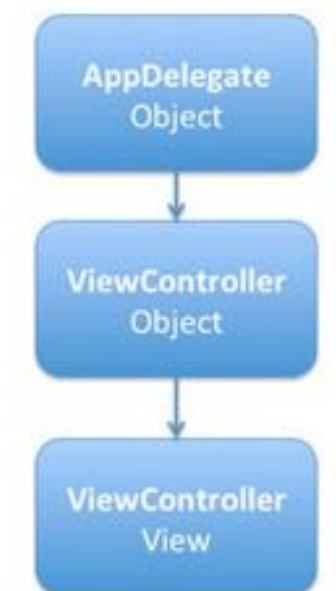


FIGURA 18.- FUNCIONAMIENTO DE LA APLICACIÓN

#### 4.3.1 Modelo-vista-controlador

El modelo-vista-controlador (MVC) es un patrón de arquitectura de software que separa los datos y la lógica de la interfaz de usuario en una aplicación. Se basa en la reutilización de código y la separación de conceptos que buscan facilitar el desarrollo de aplicaciones y su mantenimiento. En el caso particular de iOS, este patrón de desarrollo es uno de los más empleados y, por tanto, el que se va a ser la base de nuestro proyecto [10].

Los componentes de MVC se definen a continuación.

- El modelo: Es la representación de la información con la cual la aplicación opera, gestiona todos los accesos a esta información (tanto creación, actualización y borrado de la misma). Las peticiones de acceso o manipulación se reciben en el modelo a través del controlador.
- El controlador: Responde a las acciones del usuario e invoca peticiones al modelo cuando recibe una solicitud de información. Además puede enviar comandos a su vista asociada si se solicita un cambio en el modo en el que la información se muestra al usuario.

- La vista: Presenta el modelo en un formato adecuado para interactuar, más conocido como interfaz de usuario. Por tanto necesita de la información del modelo que debe representar como salida.

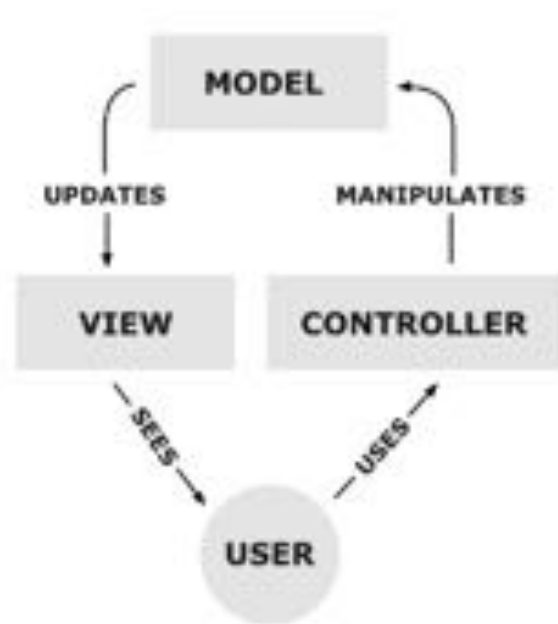


FIGURA 19.- MODELO-VISTA-CONTROLADOR

### 4.3.2 Storyboards.

El último de los grandes grupos que forman todos los contenidos del programa es el storyboard, que representa gráficamente el camino o posibles caminos que puede tomar el usuario dentro de una aplicación.

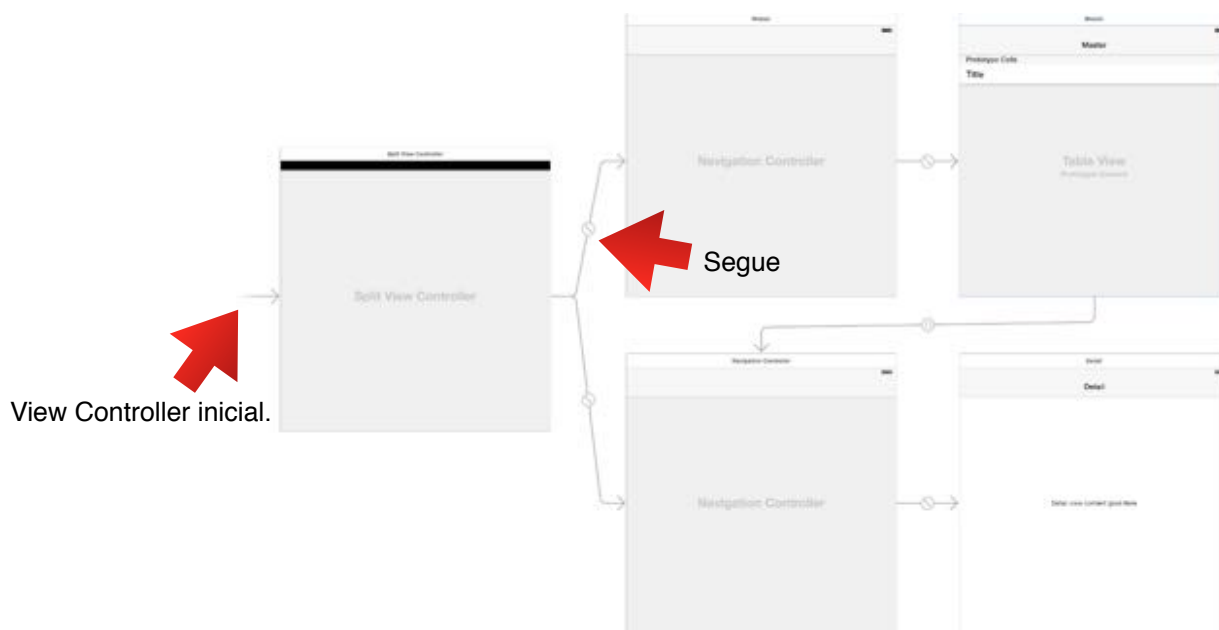


FIGURA 20.- EJEMPLO STORYBOARD


Especifica la interfaz de usuario en términos de:

- Escenas.
- Segues entre escenas.
- Controles para lanzar las segues.

La escena representa el área de contenido en la pantalla. Originalmente en el caso de iPhone, una pantalla solía contener generalmente una escena debido a la limitación de hardware. En la actualidad una pantalla puede contener varias escenas.

Una segue establece la transición de una escena a la siguiente, representado como una flecha en el Storyboard. Cada segue se asocia a un elemento específico (por ejemplo un botón) y contiene un identificador único. Dicho identificador es útil a la hora de pasar información de un ViewController a otro mediante el método *prepareForSegue* [11].

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender {  
    if ([[segue identifier] isEqualToString:@"identificador segue"])  
    {  
        DestinoViewController *vc = [segue destinationViewController];  
        [vc setMyObjectHere:object];  
    }  
}
```

 Guarda el objeto en el destino.

Las escenas más comunes en XCode se citan a continuación.





	Navigation Controller	Maneja una pila de view controllers para proporcionar una interfaz de profundización en los mismos. Habitualmente se asocia a una barra en la parte superior de la pantalla.
	Table View Controller	Recurso frecuente en todo tipo de aplicaciones, representa los datos en una lista desplazable de filas que puede ser dividida en secciones.
	Collection View Controller	Introducido en 2010 con la llegada del iPad, presenta la información mediante multitud de diseños. Es el estilo elegido en la aplicación de fotografías.
	Container View	Pueden ser del tamaño que el desarrollador prefiera, ofreciendo la posibilidad de aparecer en cualquier escena. Es un mecanismo útil en las ventanas de avisos o información.

TABLA 1.- ESCENAS DE XCODE

## 4.4 Menús.

La aplicación cuenta con 6 partes diferenciadas: Usuario, garaje, historial, documentos, matrículas y distancias. El usuario dentro de cada uno de estos apartados tendrá la posibilidad de profundizar mediante una serie de elementos (tablas, botones, etc) con los que se accede a submenús siguiendo una estructura jerárquica, permitiendo una experiencia de manejo sencilla e intuitiva.

Al ser una aplicación con varios entornos y con la finalidad de simplificar su uso, se estableció un estilo de color que predomina en todo el proyecto. Este método es el mas adecuado para potenciar el atractivo visual del proyecto. Los tres colores empleados y presentes de manera única en la aplicación son los siguientes.

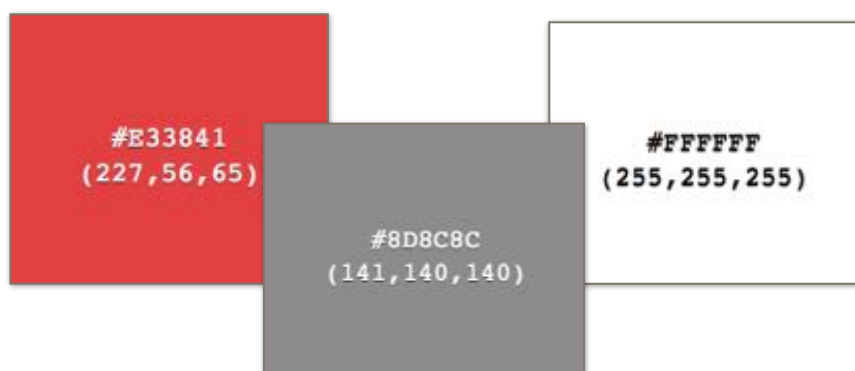


FIGURA 21.- ESTILO DE COLOR

Tras un pequeño estudio entre los usuarios de la aplicación este estilo era uno de los puntos más favorables en las valoraciones de los clientes.

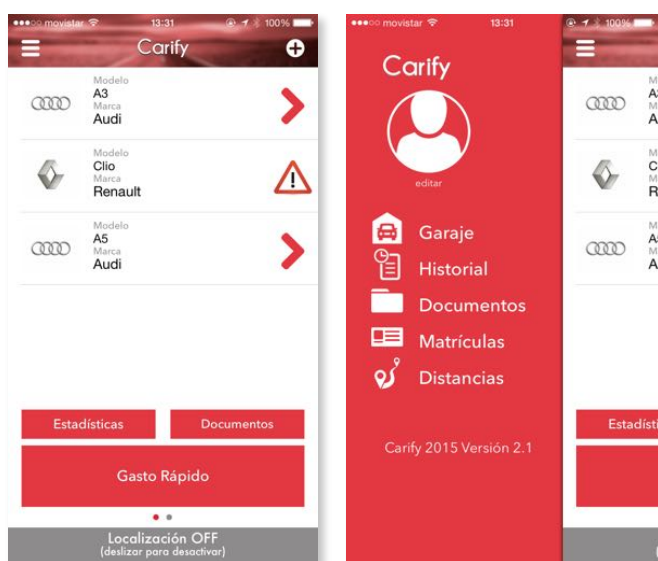


FIGURA 22A.- MENÚ DE LA APLICACIÓN

El estilo presentado es el siguiente en el menú lateral y vista principal, en principio se empleó un mayor número de colores e imágenes. Pero se descartó debido a la sensación general de falta de continuidad en la aplicación.

El resto de menús comparten el estilo de color elegido adaptado a las necesidades propias de cada apartado.

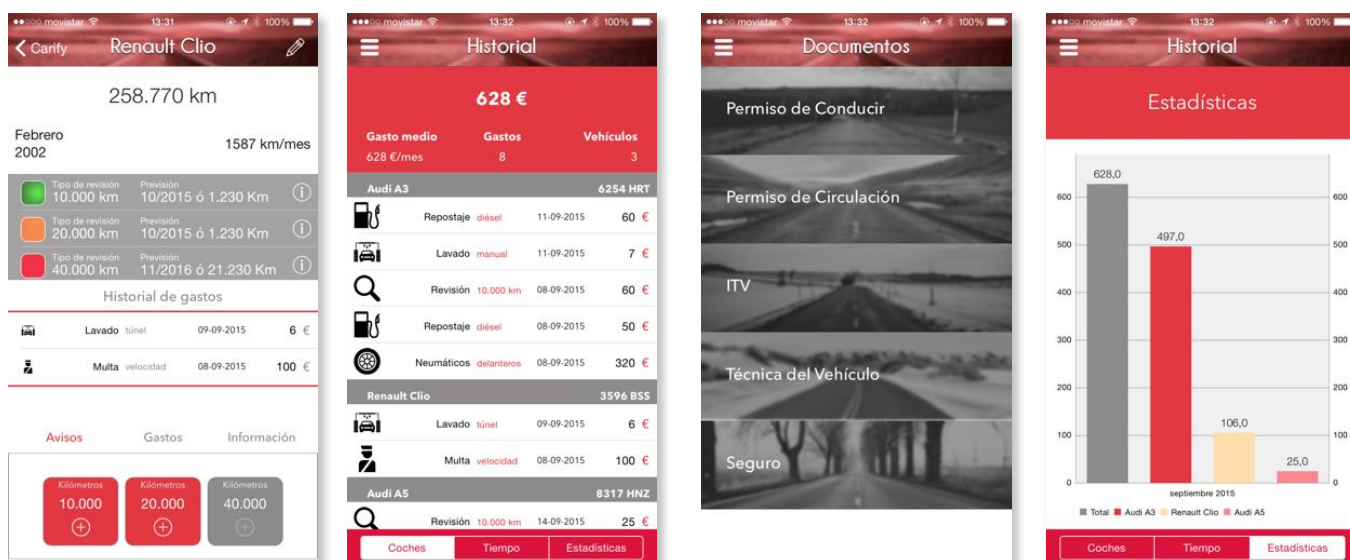


FIGURA 22B.- MENÚ DE LA APLICACIÓN

Las imágenes empleadas, así como la iconografía, fueron tratadas previamente con el software editor de gráficos vectoriales Adobe Illustrator (AI) [12].

## 4.5 Launch Screen y App icon

Continuando con el entorno gráfico, la aplicación requiere el uso de dos imágenes. La primera, conocida como launch screen, aparece durante unos segundos al lanzar la aplicación y sirve como espera mientras esta se carga. La segunda, app icon, es la imagen que el iPhone usará cuando se refiera a la aplicación, por ejemplo en los ajustes del teléfono, el botón de lanzamiento de la aplicación en el escritorio del usuario, etc.

Para estas imágenes se han empleado las líneas de estilo que adoptó el nuevo software de iOS a partir de la versión 7.0. Esta nueva línea visual se centra más en el color y en la simplicidad de la iconografía. Teniendo esto en cuenta y aplicando el estilo particular de la aplicación se crearon las imágenes con la herramienta Adobe Illustrator.



FIGURA 23.- ICONO DE LA APLICACIÓN

En el caso del icono, se optó por un diseño sencillo y claro que defina de manera concisa la esencia de la aplicación.

Para la imagen de carga de la aplicación, se partió de la base del icono y se introdujo un título para dar la bienvenida al usuario. Otra vez el objetivo del diseño es la simplicidad y atractivo visual presente de manera mayoritaria en las aplicaciones de iPhone.



FIGURA 24.- LAUNCH SCREEN

Ambas imágenes se introdujeron en el proyecto en varios tamaños para adecuarse a los distintos tamaños del dispositivo, manejando por cada imagen cinco imágenes diferentes que se cargan acorde al tipo de smartphone en el que se están ejecutando.

## 4.6 Tipografía

El objetivo principal del proyecto es la presentación de información, de modo que la claridad en la tipografía ha sido el aspecto fundamental a la hora de la elección. Tras probar primero con la tipografía por defecto de iOS se terminó por determinar otro tipo distinto que permitiera aportar a la aplicación mayor identidad propia.

La escritura finalmente elegida fue Avenir Next, diseñada por Adrian Frutiger en 2004. Actualmente es empleada en muchos sitios web y tiene numerosos reconocimientos por su particular estilo.

## 4.7 Diseño y desarrollo del garaje.

El garaje es una de las funcionalidades más importantes del proyecto y, por tanto, es la que más tiempo ha requerido en su diseño y planificación. Se compone de una herramienta compuesta por una base de datos en la que el usuario elegirá los coches que componen su garaje. Desde aquí también se puede acceder a la información específica de cada vehículo y cuenta con una serie de botones que sirven de accesos a otras secciones de la aplicación, además de un botón que permite activar la localización.

### 4.7.1 Añadir coche

La manera en la que el usuario introduce su vehículo era un aspecto determinante en el uso de la aplicación ya que es el primer contacto con la misma. En principio se optó por que toda la información relativa se introdujera mediante teclado, pero a la hora de implementarlo la experiencia de uso era lenta y de poco atractivo. Por esto decidimos crear una base de datos donde el usuario puede buscar su vehículo y seleccionarlo simplemente pulsando en la pantalla.

Para crear una base de datos con una cantidad aceptable de vehículos primero contactamos con varios sitios web especializados en esta temática, así como diversos talleres (aurgi, midas, etc). Como ninguno de ellos nos facilitó la información procedimos a realizar la base de datos desde cero.

Se crearon dos bases de datos, una con las marcas disponibles y otra con los modelos de esas marcas. Actualmente la aplicación maneja 40 marcas diferentes y 508 modelos de vehículo en total.

id	marca_vehiculo	pais_vehiculo	logo_vehiculo
1	Audi	Germany	audi.png
3	Alfa Romeo	Italy	alfaromeo.png
4	BMW	Germany	bmw.png
5	Chevrolet	USA	chevrolet.png

id	marca	modelo	logo
1	Renault	Avantime	renault.png
2	Renault	Captur	renault.png
3	Renault	Clio	renault.png
4	Renault	Duster	renault.png
5	Renault	Espace	renault.png

TABLA 2.- BASE DE DATOS SQLITE DE MARCAS Y MODELOS DE VEHÍCULO



Las bases de datos son almacenadas en arrays de memoria mediante unos métodos implementados en `manejoBaseDatos.m`. Se almacena esa información en arrays y se muestra en pantalla en una tabla, que en el entorno de programación Xcode consiste en implementar un `tableViewController`. Este permite el diseño de una celda específica que luego el programa se encarga de rellenar con la información del array y repetir las veces que sea necesario. Toda la información contenida en los arrays se elimina al cambiar de vista en la aplicación con el objetivo de reducir el uso de los recursos del teléfono.

Una vez el vehículo es elegido el usuario añade diversa información relativa, donde se encuentra uno de los pilares de todo el proyecto: La fecha de matriculación. Es importante ya que la información de kilómetros recorridos mensuales se fundamenta en este dato. Tras contactar con diversas páginas web en las que se ofrecía conocer la fecha de matriculación del coche y, ante la negativa de todas a prestar sus base de datos (incluida la dirección general de tráfico), se implementó la base de datos desde cero. La información se obtuvo de la web [sme-matriculas.es](http://sme-matriculas.es), donde se proporciona el último número de matrícula que se matriculó en cada mes.

2011	2012	2013	2014	2015**
01 E 3902 HBP	E 6656 HJC	E 0994 HNT	E 2881 HVN	E 2848 JCK

FIGURA 25.- [SME-MATRICULAS.ES](http://SME-MATRICULAS.ES)

Con esta información se crearon 174 entradas en la base de datos SQLite conteniendo el mes y el año de matriculación. Para establecer correctamente la fecha se implementó un algoritmo que recorre esta información y compara recursivamente entre dos matrículas para saber si la que ha introducido el usuario se encuentra en ese intervalo de tiempo. El algoritmo detallado se encuentra en `DetalleCocheViewController.m` en el método `touchCalcularFecha`.

Después de obtener la fecha se calcula la media mensual de distancia recorrida mediante un modelo matemático sencillo y se guarda el vehículo del usuario en la base de datos.

id	marca	modelo	matriculanum	matriculaet	matriculam	matriculay	distancia	distanciames	lastActM	lastActY
----	-------	--------	--------------	-------------	------------	------------	-----------	--------------	----------	----------

TABLA 3.- DATOS GUARDADOS DEL VEHÍCULO DEL USUARIO

Como puede apreciarse en la tabla 3, se almacena también la última fecha en la que esa distancia fue actualizada (lastActM-mes y lastActY-año). En un principio no se incluían estas entradas de datos, se añadieron posteriormente para la actualización mensual de distancia que se explicará más adelante.

#### 4.7.2 Mis coches.

En el desarrollo del apartado producido al pulsar sobre un vehículo existían muchas opciones de enfoque. En un principio se pensó como un visualizador de estadísticas. Pero tras consultar con expertos en la mecánica del automóvil se introdujeron distintos elementos que maximizan el uso de la información del vehículo que gestiona la aplicación.

El siguiente paso fue el de crear una serie de métodos que proporcionan información acerca de cuando se estiman las revisiones del vehículo, así como los elementos que deben ser revisados atendiendo al kilometraje y su coste medio. Los métodos consisten en modelos matemáticos sencillos acompañados del tipo de datos proporcionado por el entorno de programación que permite manejar las fechas almacenadas y la actual de una manera sencilla.

Como lo primordial en la aplicación es la sencillez, desde el menú principal se implementó un icono que se vuelve visible cuando no existen revisiones de ninguno de los vehículos en los próximos 3 meses. En caso contrario se mostrará el vehículo y fecha de la revisión.



FIGURA 26.- ICONO NO REVISIONES PRÓXIMAS

### 4.7.3 Localización

Para emplear la localización se incluyó la librería Core Location Manager en el proyecto. Desde la vista principal se creó un delegado de dicha librería que permite comenzar y terminar de obtener datos de localización desde el teléfono.

Parámetros del localizador.

- Nivel de exactitud: Mejor para la navegación (best for navigation).

Mide la precisión en la obtención de la localización del terminal. Puede ser la mejor posible ó con un cierto error (10 metros, 100 metros, 1 kilómetro y 3 kilómetros).

Se optó por este nivel de exactitud en la localización debido a que las otras opciones menos precisas aportaban un error de localización elevado. Con esta opción el error estimado se reduce hasta hasta un máximo de 500 metros con un consumo de batería aceptable.

- Filtro de distancia: 500 metros.

El filtro de distancia determina con que frecuencia de distancia se actualizan los datos de posición. Un mayor filtro de distancia reduce el consumo de recursos.

Tras comenzar con un filtro de 300 metros, se redujo más tarde a 500 metros ya que reducía el gasto de batería y concordaba con el posible error de precisión en la estimación de la localización.

Al recorrer una distancia mayor al filtro de distancia se accede al método `didUpdateLocations` presente en `MainViewController.m`. Este método es el encargado de medir la distancia entre ese nuevo punto de localización y el anterior obtenido y guardarlo en una variable que posteriormente se almacenará en la base de datos.

Esta herramienta además permanece activa aunque el dispositivo esté bloqueado. Esta actualización en segundo plano es posible modificando las características del proyecto, en la pestaña “Capabilities”.

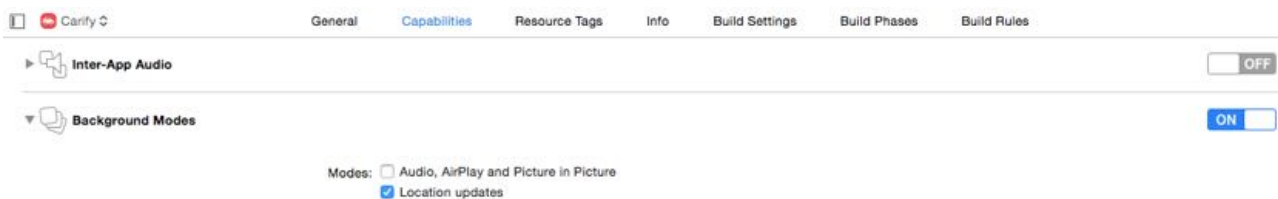


FIGURA 27.- PERMISO LOCALIZACIÓN

Bloqueando el teléfono se optimiza el uso de los recursos del mismo y no interfiere en la distancia medida finalmente.

#### 4.7.4 Coche seleccionado

Los datos específicos de cada vehículo se muestran en un layout nuevo, el cual se organiza en cuatro partes diferenciadas que se detallan seguidamente.

En la parte superior se encuentran los kilómetros recorridos por el vehículo, su fecha de matriculación y la media mensual recorrida. En principio la media mensual se pensó como un dato fijo, más adelante se incluyó la posibilidad de editarla para aumentar la personalización de los datos por parte del usuario.

Más abajo, se encuentran las fechas asociadas a los tres grandes grupos de revisión: 10.000, 20.000 y 40.000 kilómetros. Además se incluyen las fechas previstas para dichas revisiones.

En tercer lugar se muestran los gastos asociados al coche ordenados por la fecha en la que se agregaron y por último se encuentran los botones para completar las revisiones previstas. Estos botones se activan cuando existe una revisión pendiente y se desactivan al introducir el coste de la misma.

La parte inferior de la aplicación incluye información adicional en la que figura la edad del vehículo y las fechas de caducidad de los neumáticos.



FIGURA 28.- COCHE SELECCIONADO

La información modificada en esta ventana de la aplicación se almacena en la base de datos creada al añadir el coche a la biblioteca (Tabla.-3).

#### 4.7.5 Manejo de fechas.

La aplicación entera, y especialmente el apartado garaje, interactúa y procesa de manera activa multitud de fechas. Estas son de vital importancia en el correcto funcionamiento de la aplicación, especialmente en las herramientas de previsión de revisiones.

Para manejar estos datos se empleó la clase NSDate que proporciona las herramientas necesarias para comparar, representar y crear fechas. Los métodos más importantes implementados en este ámbito se citan a continuación.

Método	Función
+(NSInteger) mesesEntreDate:(NSDate*) yDate:(NSDate*)	Calcula el numero de meses existentes entre dos fechas de tipo NSDate
-(NSArray*) obtenerMesAnyo:(NSDate*) fecha	Separa la fecha en componentes de tipo NSInteger, mes y año, y las devuelve en un array.
-(NSDate*) dateFromString:(NSString*)	Recibe una instancia de NSString con el formato “día-mes-año” y lo transforma en un objeto de tipo NSDate.
-(NSArray) estimarMesesHastaRevisiónTipo:(NSString*)	Calcula el número de meses restantes para llevar a cabo una inspección en el vehículo.
-(void) actualizarDistancias	Al producirse un cambio de mes, este método actualiza la distancia con la media mensual en los vehículos del usuario y actualiza las bases de datos correspondientes.

**TABLA 4.- MÉTODOS DE MANEJO DE FECHAS**

Durante el desarrollo del garaje surgió un problema en la gestión de revisiones en el que, cuando se completaba la revisión antes de vencer la fecha estimada y se recargaba la aplicación, el código empleado no reconocía esa revisión como terminada y retornaba el aviso de revisión pendiente.

Para solucionar este problema se optó por la creación de una nueva entrada en la base de datos con el nombre de revisiones, con el formato siguiente.

entrada	id	idCar	tipo	próxima	terminada	fecha
definición	identificador de la entrada en la tabla	coche asociado a la revisión	tipo de revisión	revisión en menos de tres meses	revisión finalizada	fecha de finalización de la revisión
ejemplo	1	2	40000	yes	no	09-2015

**TABLA 5.- TABLA REVISIONES DE SQLITE**

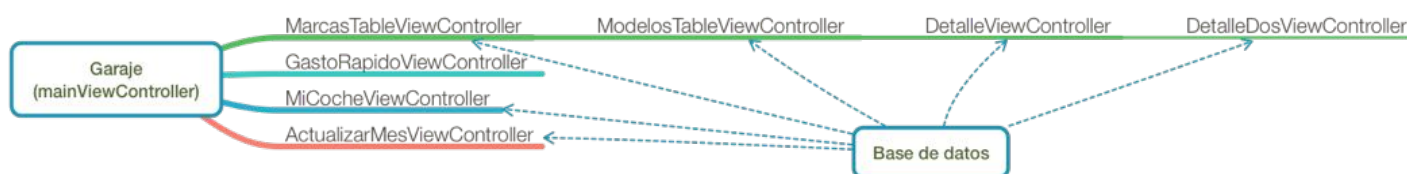
Una vez el usuario completa una revisión, la entrada en la base de datos correspondiente a su tipo se actualiza con la fecha correspondiente a la finalización y el campo terminada pasa a ser “yes”.

Al reiniciar la aplicación esta obtiene las entradas de información de revisiones y muestra avisos en función de si las revisiones están terminadas. Una vez la fecha actual supera a la de la revisión prevista la entrada en revisiones se actualiza estableciendo los campos próxima y terminada a “no”.

Esta solución se comprobó como la más robusta a posibles errores y fue la solución final tomada en el desarrollo de la aplicación.

#### 4.7.6 Esquema del apartado.

A continuación se muestra un esquema general del módulo Garaje.



**FIGURA 29.- ESQUEMA DE GARAJE**

#### 4.7.7 Descripción de archivos.

Nombre	Descripción
mainViewController	Da la opción al usuario de elegir cualquier coche dentro de su garaje, así como de añadir coches.
MarcasTableViewController	Tabla que permite seleccionar la marca del coche nuevo a introducir en el garaje del usuario.
ModelosTableViewController	Modelos existentes de la marca seleccionada por el usuario previamente al añadir un coche.

DetalleViewController	Al elegir la marca y modelo del nuevo vehículo, este archivo se encarga de gestionar la matrícula y kilómetros recorridos introducidos por el usuario.
DetalleDosViewController	Visor de todas las opciones elegidas por el usuario en el vehículo. Al aceptar estos datos el nuevo vehículo se añadirá a la base de datos.
GastoRapidoViewController	Modelos existentes de la marca seleccionada por el usuario previamente al añadir un coche.
MiCocheViewController	Archivo encargado de mostrar el historial de gastos, próximas revisiones y demás información del coche seleccionado.
ActualizarMesViewController	Al comenzar un nuevo mes muestra los kilómetros totales calculados mediante la herramienta de localización de cada vehículo.

**TABLA 6.- ARCHIVOS DEL MÓDULO GARAJE**

## 4.8 Diseño y desarrollo de historial.

Tanto la parte correspondiente al garaje como al historial de gastos del usuario se fundamenta en el manejo de bases de datos SQLite. La herramienta adicional que proporciona el historial consiste en un visor de estadísticas mediante gráfico de barras que se explicará más adelante.

### 4.8.1 Base de datos.

Una base de datos puede contener un número ilimitado de tablas, cada una de ellas con identificador diferente. En cada una de las columnas se organizan los datos de forma diferente.

Para la gestión del historial, se aprovechó el archivo de tipo sql creado para manejar los datos del garaje y se introdujo una nueva tabla que gestionaría el historial de gastos del cliente. La estructura de datos de la tabla se pensó originalmente de la siguiente manera.

id (autoincrementable)	matricula	tipo	detalle	precio	fecha
1	6547BDS	repostaje	diesel	50	21-09-1990

**TABLA 7.- TABLA HISTORIAL INICIAL**

Con esta organización de la información se aprovechaba la matrícula del vehículo como identificador a la hora de obtener la información relacionada a un vehículo en concreto mediante la creación de varios métodos en manejoBaseDatos.m.

Este modelo se volvió problemático a la hora de agrupar la información del historial tanto por fecha como por vehículo. Los métodos de obtención de información del historial implementados mediante filtros resultaban demasiado extensos para la tarea a ejecutar y demoraban en general el funcionamiento de la aplicación.

La solución final empleada para mejorar el rendimiento general fue la de obtener toda la información de las entradas del historial en un array, para luego establecer métodos de ordenamiento de dichas entradas. Además, se eliminó el campo matrícula de la base de datos y se empleó como identificador del coche asociado a ese gasto la variable asignada de manera automática en la tabla de vehículos de usuario. Esta variable no se repite en ningún caso por lo que se trataba de un identificador sencillo y eficaz.

La estructura de la base de datos finalmente se ordenó como sigue.

id	clavevehiculo	tipo	detalle	precio	moneda	fecha
1	2	repostaje	diesel	50	euros	21-09-2015

**TABLA 8.- TABLA HISTORIAL FINAL**

Como puede apreciarse en la tabla anterior, se añadió el campo moneda como una posible mejora futura de cambio de divisas.

#### **4.8.2 Edición del historial.**

Aparte de ordenar y mostrar la información al usuario era necesario añadir una herramienta que permitiese la eliminación de aquellas entradas del historial que se han introducido erróneamente o que el usuario no desee conservar.

En la fase de diseño del proyecto se implementó esta posibilidad mediante un pulso de larga duración en la celda de la tabla a eliminar. La implementación del reconocedor de pulsos largos se obtuvo mediante la clase `UILongPressGestureRecognizer`, esta se inicializa en el código del proyecto y se le asigna un método determinado que se lanza cada vez que se reconoce una pulsación larga en la pantalla del smartphone.

Tras comprobar el funcionamiento de la aplicación en un iPhone esta opción se descartó debido al pequeño tamaño de las celdas, haciendo un tanto engorrosa la experiencia del usuario al ser pulsadas.



Tras un estudio de los posibles interfaces de eliminación presentes en las aplicaciones de gestión de historiales similares finalmente se optó por las celdas desplazables. Este tipo de interfaz es de los más empleados en la actualidad y añade una mejora al estilo general de la aplicación.

El funcionamiento básico de la interfaz consiste en desplazar la celda hacia la izquierda, desplegando un botón oculto que da la posibilidad de eliminar la entrada del historial.



FIGURA 30.- ELIMINAR ENTRADA DEL HISTORIAL

Para implementar esta función se empleó la librería `SWTableViewCell` desarrollada por Chris Wendel. Esta librería forma una subclase de `UITableViewCell` que se encarga de dar formato a las celdas en las tablas mostradas en el smartphone. Instanciando esta librería se pueden añadir tantos botones como el desarrollador necesite.

Para prevenir el borrado de entradas no intencionado por parte del usuario se introdujo una vista de alerta que es lanzada siempre que el usuario presione el icono de la papelera y que informa al usuario del borrado de la información en la base de datos.



FIGURA 31.- ALERTA HISTORIAL

Por cada nueva aparición en pantalla del historial el array se actualiza con toda la información de los vehículos. La eliminación de entradas se realiza mediante métodos de búsqueda en la base de datos, con la consiguiente actualización del array de información.

### 4.8.3 Estadísticas.

Para finalizar el desarrollo del historial se introdujo la posibilidad de ver todos los gastos mediante un gráfico de barras que se actualiza con el tiempo, dicho visor permite centrar la vista en uno o varios meses mediante gestos del propio usuario.

Como primera medida a la hora de implementar este nuevo módulo se diseñó como una vista dentro de la ventana, y no como otra ventana nueva, debido al gran número de ventanas que ya manejaba el programa y que podría afectar el rendimiento de la aplicación.

Además las herramientas proporcionadas por Apple para la creación de gráficos fueron desarrolladas recientemente, por lo que el lenguaje de programación era distinto (Swift). Este nuevo lenguaje ofrece la posibilidad de asociarse con proyectos escritos en Objective-C.

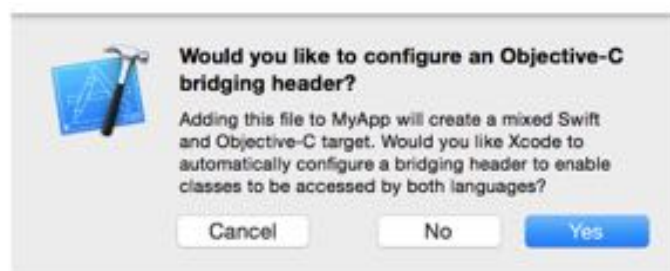


FIGURA 32.- INTRODUCIR SWIFT EN PROYECTO OBJECTIVE-C

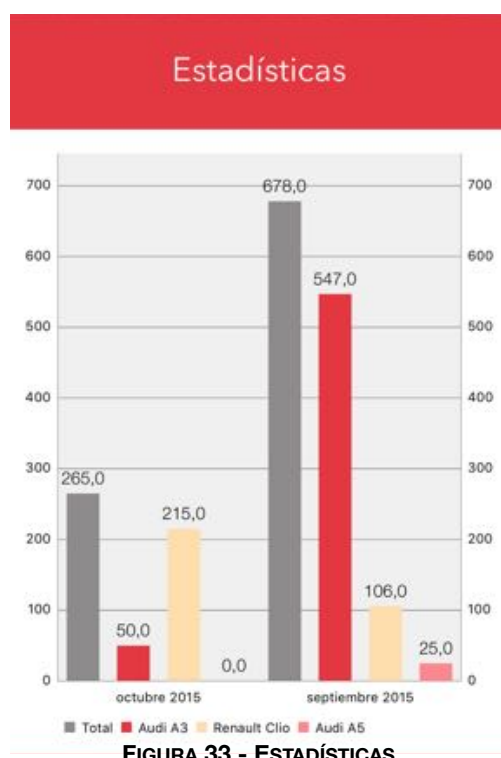


FIGURA 33.- ESTADÍSTICAS

Una vez introducido el archivo en código Swift, Xcode crea un archivo .h que sirve de puente entre los dos lenguajes. Posteriormente, realizando varios cambios en los parámetros generales del proyecto e introduciendo la clase en el archivo donde se va a emplear esta librería ya posibilita la coexistencia de ambos códigos.

Para la ordenación correcta de la información se implementó un método consistente en la agrupación de información en un array doble que organizase primero los gastos en meses para luego filtrar la información según el vehículo al que estaba asignada.

#### 4.8.4 Esquema general del módulo.

A continuación se muestra una figura que representa el funcionamiento del apartado historial dentro de la aplicación.



FIGURA 34.- ESQUEMA HISTORIAL

#### 4.8.5 Descripción de archivos.

Nombre	Descripción
HeaderTableViewCell	Archivo encargado de dar formato a la celda principal de la tabla donde se encuentra el número de gastos total, número de coches y media de gasto mensual en los vehículos.
HistorilViewController	Gestiona las tres vistas que ofrece el módulo e intercambia dichas vistas según el botón pulsado por el usuario en la parte inferior.
HistorialTableViewCell	Subclase de SWTableViewCell, se encarga de proporcionar la interfaz a las celdas del módulo e introduce los botones desplazables.

TABLA 9.- ARCHIVOS DE HISTORIAL

#### 4.9 Diseño y desarrollo de documentos.

El módulo documentos es la funcionalidad secundaria más importante en la aplicación. En ella se pueden almacenar imágenes y archivos relacionados con la documentación del vehículo y del propio usuario.

Tras consultar sobre la documentación obligatoria que es preciso llevar en cualquier vehículo se establecieron cinco categorías en las que organizar los archivos: Permiso de conducir, permiso de circulación, inspección técnica de vehículos, técnica del vehículo y seguro. Cada categoría ofrece un lugar donde guardar los archivos y ofrece varios métodos de carga de archivos, aprovechando las distintas herramientas que posee el smartphone.

### 4.9.1 Manejo de archivos

Desde una primera instancia se incluyó en la aplicación la posibilidad de cargar imágenes desde el carrete y la cámara integrada en el dispositivo. Por lo que la primera parte del desarrollo del módulo se centró en investigar sobre estas funcionalidades.

La herramienta visual elegida para incluir en la interfaz encargada de mostrar los documentos fue la conocida como collection view. Dicha ventana se encarga de mostrar un conjunto ordenado de datos que se presentan en pantalla de una manera flexible, ofreciendo la posibilidad de ofrecer miniaturas de los archivos que maneja.



FIGURA 35.- EJEMPLO COLLECTION VIEW

Junto con esta ventana, se añaden dos botones en la parte inferior de la misma que se encargarán de gestionar la cámara y el carrete respectivamente. Ambas implementaciones son similares y hacen uso de la clase UIImagePickerControllerController siguiendo la pauta detallada a continuación.

```
#pragma mark Manejo Items Bottom
- (IBAction)cameraItemTouched:(id)sender {
    UIImagePickerController *picker=[[UIImagePickerController alloc] init];
    picker.delegate=self;
    picker.allowsEditing=false;
    picker.sourceType=UIImagePickerControllerSourceTypeCamera;

    [self presentViewController:picker animated:true completion:NULL];
}
```

FIGURA 36.- MANEJO DE LA CÁMARA

Al tomar la fotografía o seleccionar una del carrete se cargará directamente en la carpeta documentos de la aplicación. En un principio se almacenaban las fotografías en el directorio donde se almacenaban los iconos de la aplicación pero repercutía negativamente en el uso de memoria por parte de la aplicación.

Las imágenes almacenadas soportan el formato JPEG y PNG.

Para eliminar los archivos se empleó un reconocedor de pulsaciones de larga duración en la que se muestra un aviso por pantalla en el que informa del borrado del archivo. En caso de aceptar la operación el archivo se elimina de la carpeta y se actualiza el array encargado de listar los archivos presentes en la carpeta.

#### 4.9.2 Sincronización de archivos.

Para ofrecer mayor rango de posibilidades al usuario la aplicación ofrece posibilidad de descargar los archivos almacenados en la nube para visualizarlos en la aplicación. Tras un estudio sobre las múltiples opciones en el mercado actual se optó por el empleo de Dropbox ya que es la más popular y, por tanto, la que mejor margen de penetración en el mercado presenta.

La API<sub>[1]</sub> de Dropbox se puede descargar de manera gratuita en su página web, en ella se encuentran tutoriales de aprendizaje sobre las herramientas que ofrece y las técnicas para implementarlas. Existen dos maneras básicas de gestionar una aplicación en Dropbox: Mediante el uso de una carpeta ó con acceso libre a todos los documentos del usuario. Por motivos de seguridad se prefirió crear una carpeta donde se almacenase todos los archivos a usar en la aplicación.

Con esta información completada el sistema de almacenamiento en la nube proporciona al desarrollador dos claves que deben introducirse en los parámetros generales del proyecto.

Permission type	App folder ⓘ
App folder name	TutApp
App key	17wha...
App secret	[blurred]

FIGURA 37.- PARÁMETROS APLICACIÓN EN DROPBOX

Los métodos principales implementados para la correcta interacción con el sistema de almacenamiento de citan a continuación junto con una breve descripción de los mismos.

Método	Descripción
<code>initDropboxRestClient</code>	Comprueba si el usuario ha vinculado su cuenta de Dropbox a la aplicación y en caso contrario muestra un mensaje y la posibilidad de vincularse.
<code>uploadProgress</code>	Ofrece el porcentaje de progreso en la subida de un archivo a la nube.
<code>uploadedFile</code>	Al terminar la subida se encarga de avisar al usuario.
<code>loadedMetadata</code>	Método encargado de listar los contenidos de un directorio en Dropbox.

**TABLA 10.- MÉTODOS DE INTEGRACIÓN CON DROPBOX**

Al terminar la implementación del módulo de sincronización de archivos y comprobado su correcto funcionamiento se incluyó la capacidad de descargar archivos ya existentes en la nube en la aplicación. Los métodos empleados son similares a los citados en la tabla anterior pero en este caso respecto a la bajada de archivos.

Al añadir esta mejora se desarrolló el código necesario para que la aplicación soportase la visualización de PDF debido a la gran presencia de documentos en este formato.

#### 4.9.3 Esquema general del módulo.

A continuación se muestra una figura que representa el funcionamiento del apartado documentos dentro de la aplicación.



**FIGURA 38.- ESQUEMA MÓDULO DOCUMENTOS**

#### 4.10 Diagrama de clases de la aplicación.

Seguidamente se ofrece un diagrama completo de la aplicación, con todas sus clases, métodos, variables y relación entre ellos.

[illegible]

## 5 PRUEBAS Y RESULTADOS

### 5.1 Publicación en Apple App Store.

Habiendo finalizado el desarrollo de la aplicación y realizado pruebas en varios dispositivos para comprobar su funcionamiento, se solicitaron evaluaciones de la aplicación tanto a expertos en mecánica automovilística como a conductores con conocimientos básicos sobre vehículos. Ambos grupos de mercado valoraron positivamente la aplicación de modo que el siguiente paso fue la publicación de la aplicación en el App Store.

La aplicación se subió a la cuenta de desarrollador creada por el director del proyecto Pablo Zarco, siguiendo los pasos detallados a continuación. La plataforma ofrecida por el sistema para gestionar aplicaciones se denomina iTunes Connect, esta provee a los desarrolladores de diversas herramientas de análisis de ventas y ayuda.

1. Creación de un certificado de distribución: Este certificado que se almacena en el perfil del desarrollador confirma la identidad de este y establece una firma en su código. Una vez creado se añade al acceso de llaveros del ordenador, siendo solo necesario un permiso para todas las aplicaciones que se deseen subir.

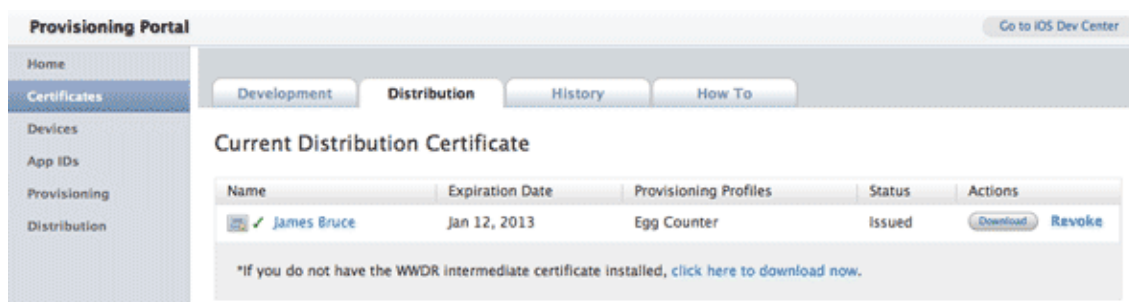


FIGURA 40.- CERTIFICADO DE DISTRIBUCIÓN

2. Creación de un perfil provisional de distribución: Siguiendo con los certificados necesarios para publicar aplicaciones, el siguiente consiste en un archivo de tipo CSR que contiene información acerca del entorno de publicación (ordenador, smartphone, reloj...). Al obtener el archivo se incluirá en los parámetros generales del proyecto.
3. Subida de compilación de la aplicación: Desde el entorno de programación XCode, habiendo comprobado el correcto funcionamiento de la aplicación, se subirá el

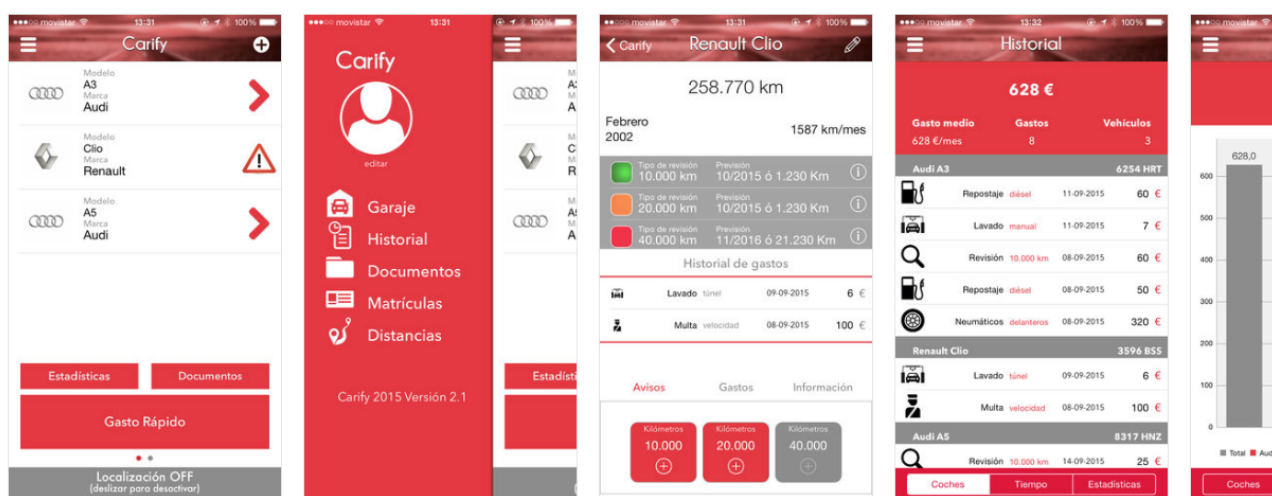


Introducción de información: Tras completar la gestión de certificados y compilación los pasos siguientes se realizan vía web. Debemos rellenar todos los campos de información relativos a nuestra app. Contenidos como nombre, capturas de pantalla de los dispositivos en los que se va a emplear, descripción, etc.

Español (España) ▾ ?

Vista previa del vídeo y capturas de pantalla de la app ?

4,7 pulgadas	5,5 pulgadas	4 pulgadas	3,5 pulgadas
--------------	--------------	------------	--------------



Descripción ?

Con Carify podrá llevar a cabo un seguimiento integral de todos sus vehículos.

¡Introduzca su coche y la aplicación se encargará de calcular su media mensual de kilómetros recorridos y avisarle de las próximas revisiones del vehículo!

Si desea usar el localizador, actívelo y guarde la distancia recorrida en un trayecto en el iPhone. Al final todas esas distancias se añadirán a su kilometraje.

### Características de la App.

- Almacenaje opcional de documentación del vehículo.
- Vinculación con Dropbox.
- Visualización de estadísticas de gastos.
- Avisos de revisiones pendientes.
- Calculador de fecha de matriculación.

Aviso! las matrículas contempladas en la app son del tipo 0000 BBB

Palabras clave ?


coche, vehículo, car, gestion, gestión, análisis, dinero, money, tracker

URL de soporte ?

<http://www.google.com>

URL de marketing ?

## Compilación

Compilación	Fecha de carga
 1.0 (1)	El 6 de oct. de 2015 a las 14:39

#### FIGURA 41.- INFORMACIÓN DE LA APLICACIÓN

5. Tras enviar la aplicación compilada con todos los datos es revisada por los desarrolladores de la empresa para comprobar un mínimo de prestaciones, rendimiento y utilidad de la aplicación. El tiempo de espera medio en llevar a cabo este proceso es de 7 días.
6. Una vez superado el examen, la aplicación se publica en el mercado de aplicaciones de manera automática.



**FIGURA 42.- APP LISTA PARA VENDER**

El precio de salida de la aplicación se estableció en 0.99 €, que se desglosan en un 70% de los ingresos por la aplicación para el grupo desarrollador y un 30% para la empresa Apple.

## **5.2 Estadísticas de la aplicación.**

Debido a la reciente publicación de la aplicación, los datos de análisis de ventas de la misma aún no se encuentran disponibles en iTunes Connect. Aún así la aplicación ha sido descargada 10 veces en un intervalo corto de tiempo, de modo que se espera su buen funcionamiento en el mercado. Según su evolución se estudiará una bajada del precio de manera temporal o el paso a un modelo gratuito con funcionalidades de pago.

Además se introducirá la aplicación en varios foros dedicados a automóviles para conseguir un mayor grado de penetración en el mercado. Así como se trabajará con diversos talleres con el objetivo de promocionar la aplicación entre sus clientes y expandir el número de potenciales usuarios.

# 6 CONCLUSIONES Y TRABAJO

---

## 6.1 Conclusiones

La realización completa del proyecto ha cumplido los objetivos establecidos en un principio. Además, conforme a la mejor comprensión del entorno de programación y las herramientas disponibles, se han desarrollado funcionalidades adicionales que no estaban previstas y que mejoran sensiblemente el proyecto.

El objetivo inicial de la aplicación era el de crear una aplicación que monitorizara todas aquellas actividades relacionadas con la gestión y mantenimiento del vehículo, incluyendo una base de datos de vehículos en las que el cliente podría seleccionar el suyo propio de manera sencilla.

La aplicación final presenta las siguientes funcionalidades:

- Registro de vehículos de usuario en la base de datos, con información editable por el propio usuario.
- Avisos de revisiones próximas en el vehículo tanto por tiempo estimado como por kilómetros restantes. En estas revisiones además se ofrece información sobre los aspectos más importantes a tener en cuenta según su kilometraje.
- Previsiones de recambio de neumáticos, actualizables en todo momento por el usuario.
- Visualización del historial de gastos del usuario, incluyendo varias posibilidades de interfaz (coches, meses y estadísticas). Posibilidad de eliminación de dichas entradas de datos.
- Calculador de fecha de matriculación de vehículos mediante la creación por parte del desarrollador de una base de datos.
- Introducción de localización para permitir un seguimiento más detallado de las distancias recorridas por los vehículos.
- Gestión de documentos relacionados por el vehículo. Gestión de varios tipos de archivos (JPEG, PNG y PDF).
- Sincronización con el sistema de almacenamiento en la nube Dropbox, ofreciendo la posibilidad de subida y bajada de archivos.
- Uso de la cámara y el carrete del dispositivo móvil para obtener documentos.

Teniendo en cuenta el aspecto didáctico, el objetivo a completar era el de aprender a desarrollar aplicaciones en entorno de programación iOS desde cero. Los objetivos logrados durante el desarrollo de la aplicación se listan a continuación.

- Programación Objective C y Swift.
- Manejo de bases de datos SQL.
- Manejo de software de tratamiento de imágenes Adobe illustrator.
- Manejo de software de sincronización con la nube Dropbox API.
- Aprendizaje de la arquitectura iOS.

## **6.2 Trabajo Futuro.**

Durante el desarrollo y finalización del proyecto, especialmente en el testeo de la aplicación, se han detectado posibles mejoras de la aplicación que serán introducidas en nuevas versiones del software.

Mejoras en el desarrollo.

- Gestión de motocicletas y camiones en la aplicación.
- Integración de un hardware específico que permita a la aplicación conectarse mediante bluetooth a vehículos que dispongan de esta tecnología y obtener sus estadísticas (kilómetros, próximas revisiones, etc).
- Compartir información en redes sociales.
- Manejo de diversas opciones de almacenamiento en la nube aparte de Dropbox.
- Añadir más vehículos a la base de datos.

Mejoras en el diseño

- Inclusión de más interfaces de visualización de estadísticas.
- Mejoras gráficas en la aplicación.
- Posibilidad de rotación de pantalla.

Mejoras en la publicidad.

- Incorporación de anuncios en páginas web especializadas.
- Versión gratuita con funcionalidades de pago.
- Inclusión de la aplicación en redes sociales.

## 7 MANUAL DE USO DE LA APLICACIÓN

---

El primer paso antes de usar la aplicación consiste en buscarla en el mercado de aplicaciones de Apple, una vez instalada ya podremos ejecutar la aplicación.

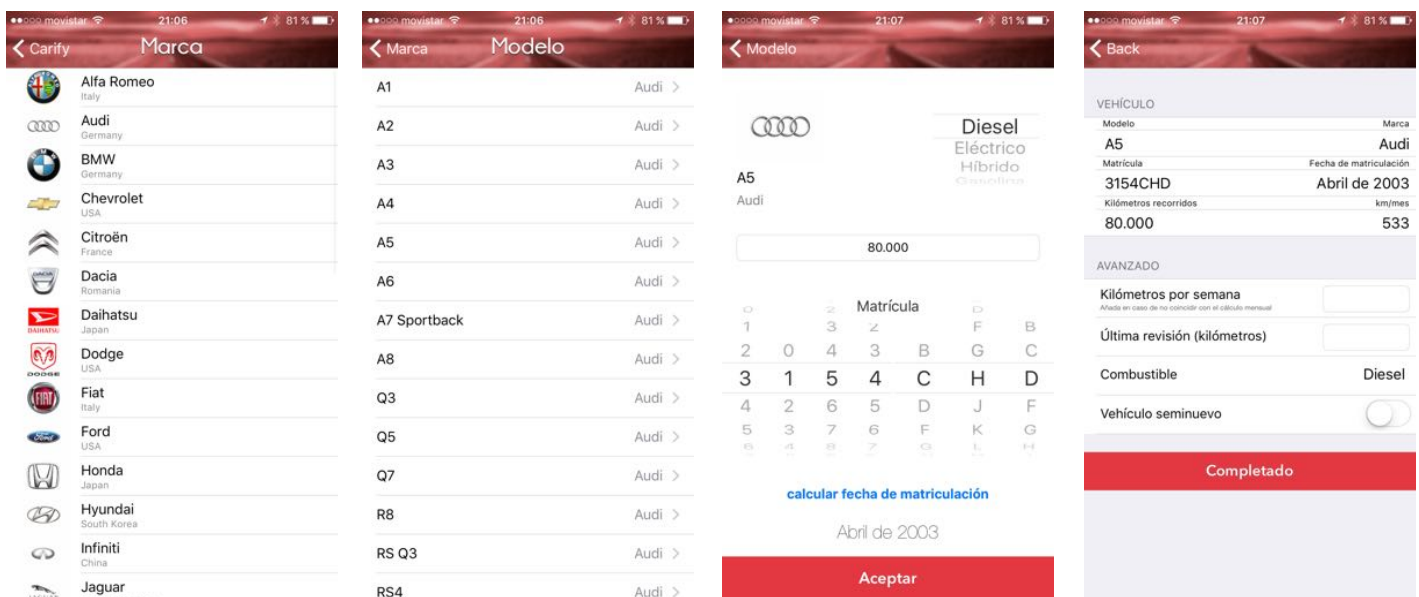
Lo primero que aparecerá en pantalla será la imagen de inicio o Launch Screen descrita en el apartado 3 de diseño de la aplicación. Esta permanecerá unos segundos en pantalla mientras se carga la interfaz de la aplicación.



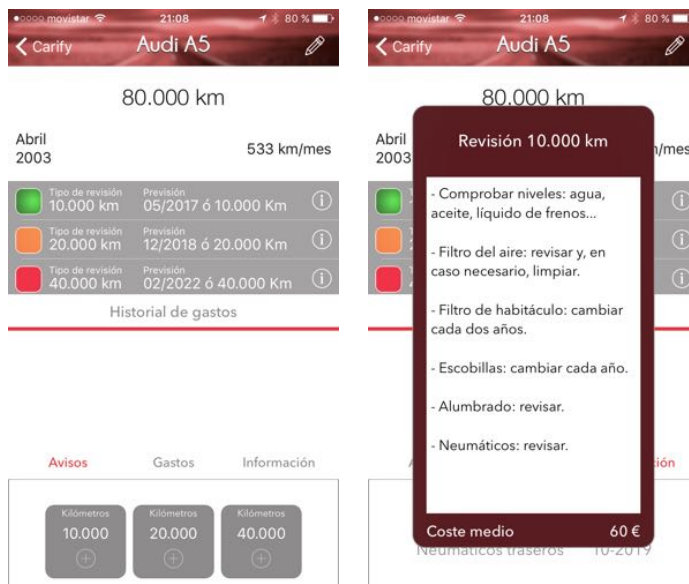
La vista inicial de la aplicación es el garaje del usuario, por defecto se encuentra vacío. Para añadir coches, el usuario deberá seleccionar el botón “+” presente en la parte superior derecha de la ventana.



Para añadir un coche el usuario deberá elegir, en tres pasos, la marca, modelo y detalles del nuevo vehículo. En el último paso se añadirá la matrícula del vehículo y la distancia recorrida hasta ese momento.



Una vez creado el vehículo, el usuario podrá acceder a información específica del mismo. En este nuevo menú se puede acceder a información sobre la previsión de revisiones e información acerca del vehículo.



Al pulsar sobre los botones de información una ventana animada aparecerá en la ventana conteniendo información sobre la revisión seleccionada y el gasto medio de la misma. Pulsando fuera de la ventana desaparecerá la vista mediante una animación.

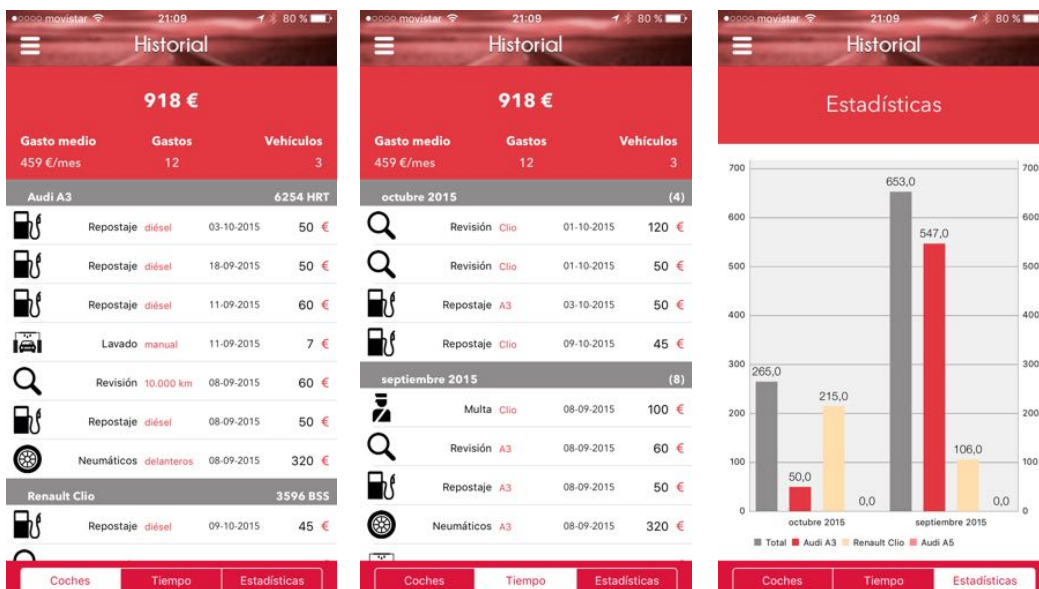
Cuando una revisión pendiente se encuentre a menos de tres meses de la fecha actual los botones de la parte inferior de la ventana se activarán para completar la misma. Una vez introducidos los datos se almacenará una entrada en el historial con la revisión.

Para introducir un gasto existe un botón disponible en el menú principal en el garaje donde se muestran varios campos de información a completar por el usuario. Al pulsar el botón aceptar, si los datos se han introducido de manera correcta, se asocia dicha información al coche y se almacena en la base de datos.



Para visualizar el historial del usuario primero se pulsará el botón de menú situado en la parte superior izquierda de la pantalla. El menú se despliega en la parte izquierda de la ventana, desde el que se accede a todas las herramientas de la aplicación.

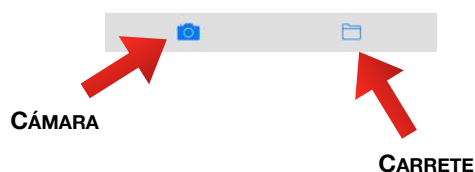
Una vez seleccionado el historial se ofrecen tres posibles interfaces para visualizar los datos del usuario: Agrupado en meses, agrupado por coches y un gráfico de barras. El visor de gráficos ofrece la posibilidad de acercar y alejar las mismas mediante gestos de zoom.



Para la gestión de documentos se deberá pulsar primero el menú desplegable mencionado anteriormente. Una vez pulsada la sección documentos se despliegan en pantalla cinco carpetas donde almacenar las imágenes o pdf del usuario.



Una vez el cliente seleccione el lugar deseado existen tres posibilidades de importar datos: Mediante la cámara, a través del carrete de fotografías del smartphone y mediante Dropbox.



En caso de pulsar Dropbox y no tener vinculada una cuenta, la aplicación solicitará los datos de acceso del usuario para vincular su almacenamiento en la nube.



Una vez cargados los documentos en la aplicación se ofrecerá una vista en miniatura de estos. Pulsando sobre las miniaturas el programa ofrecerá una visión a pantalla completa de los archivos. En caso de tener una cuenta de Dropbox vinculada se ofrecerá de subir el documento a la nube en caso de que no exista previamente.



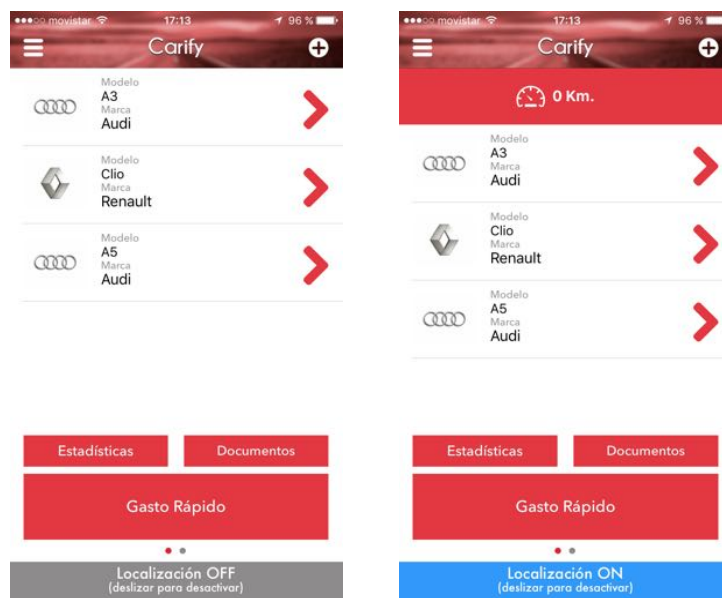
Además el visor de documentos ofrece la posibilidad de cambiar el nombre del archivo pulsando en el icono de la parte superior derecha de la pantalla.

El apartado matrículas del menú lateral muestra una ventana donde se puede calcular rápidamente la fecha de matriculación de cualquier vehículo. Además, muestra un historial de las últimas cinco matrículas calculadas por el usuario.

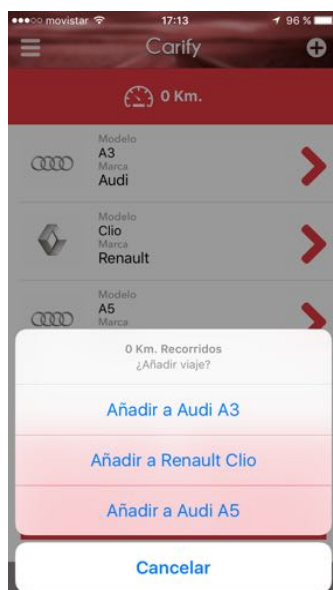


## Uso de localización.

Antes de comenzar un trayecto en coche, para usar el localizador del smartphone el usuario deberá deslizar a la derecha la barra presente en la parte inferior de la pantalla del apartado garaje. El programa mostrará una ventana con la distancia del vehículo recorrida.



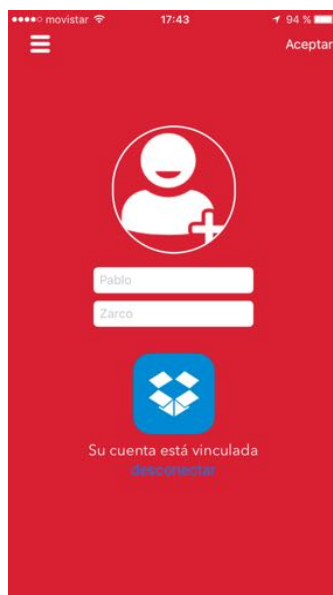
Al finalizar el trayecto el usuario deberá deslizar la misma barra en sentido contrario, la aplicación mostrará la distancia recorrida y pedirá al usuario seleccionar el vehículo con el que se ha realizado el trayecto. El historial de distancias recorridas puede consultarse a través del menú lateral pulsando en “Distancias”.



Cuando se produzca un cambio de mes, la aplicación mostrará una ventana en la que el usuario podrá decidir añadir los kilómetros recorridos mediante la suma de distancias obtenida mediante localización, o añadir la media calculada por la aplicación. Dicha elección es individual para cada vehículo.

## Usuario

Para introducir información sobre el usuario de la aplicación puede pulsarse el icono de usuario presente en el menú lateral. Puede introducirse información (nombre, apellidos, fotografía) del usuario, así como de la vinculación de una cuenta de Dropbox.



# REFERENCIAS

---

[1] <http://smartphoneavancetecnologico.blogspot.com.es/p/historia-y-evolucion-del-smartphone.html>

[2] <http://www.xatakamovil.com/mercado/desarrollo-de-aplicaciones-moviles-i-asi-esta-el-mercado>

[3] <http://www.altag.net/6-causas-por-las-que-una-aplicacion-fracasa/>

[4] <http://www.androidauthority.com/google-play-store-vs-the-apple-app-store-601836/>

[5] <https://developer.apple.com/xcode/>

[6] <https://www.sqlite.org>

[7] Simon Ng. “Learn iOS 7 programming from scratch”, Appcoda.

[8] <https://www.dropbox.com/developers>

[9] <https://github.com/danielgindi/ios-charts>

[10] <https://es.wikipedia.org/wiki/Modelo-vista-controlador>

[11] Stephen G. Kochan. “Programming in Objective-C (Developer's Library)”, Fifth Edition.

[12] Stack Social. “iOS Designer Bundle”

# PRESUPUESTO

---

## 1) Ejecución Material

• Compra ordenador personal	1.200 €
• Programa desarrolladores de Apple	99 €
• iPhone 6	699 €
• Material de Oficina	40 €
• Total ejecución material	2.038 €

## 2) Gastos generales

• 16 % sobre Ejecución Material	326 €
---------------------------------	-------

## 3) Beneficio Industrial

• 6 % sobre Ejecución Material	122 €
--------------------------------	-------

## 4) Honorarios Proyecto

• 700 horas a 15 € / hora	10.500 €
---------------------------	----------

## 5) Material fungible

• Gastos de impresión	60 €
• Gastos de encuadernación	200 €

## 6) Subtotal del presupuesto

• Subtotal	13.246 €
------------	----------

## 7) I.V.A aplicable

• 21 % Subtotal Presupuesto	2.781,66 €
-----------------------------	------------

## 8) Total presupuesto

• Total presupuesto	16.027,66 €
---------------------	-------------

Madrid, Noviembre de 2015  
El Ingeniero Jefe de Proyecto  
Fdo.: Pablo Zarco Sanz  
Ingeniero de Telecomunicación

# PLIEGO DE CONDICIONES

---

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de la siguiente aplicación: “Aplicación iOS de gestión mecánica de vehículos”. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

## **Condiciones generales**

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero

Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partidaalzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.



20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad “Presupuesto de Ejecución de Contrata” y anteriormente llamado “Presupuesto de Ejecución Material” que hoy designa otro concepto.

### **Condiciones particulares**

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.